Extracting Symptoms and their Status from Clinical Conversations

Nan Du[*], Kai Chen, Anjuli Kannan, Linh Tran, Yuhui Chen, and Izhak Shafran

{dunan, kaichen, anjuli, tranlm, yuhuic, izhak}@google.com

Google Inc.

Abstract

This paper describes novel models tailored for a new application, that of extracting the symptoms mentioned in clinical conversations along with their status. Lack of any publicly available corpus in this privacy-sensitive domain led us to develop our own corpus, consisting of about 3K conversations annotated by professional medical scribes. We propose two novel deep learning approaches to infer the symptom names and their status: (1) a new hierarchical span-attribute tagging (SA-T) model, trained using curriculum learning, and (2) a variant of sequence-to-sequence model which decodes the symptoms and their status from a few speaker turns within a sliding window over the conversation. This task stems from a realistic application of assisting medical providers in capturing symptoms mentioned by patients from their clinical conversations. To reflect this application, we define multiple metrics. From inter-rater agreement, we find that the task is inherently difficult. We conduct comprehensive evaluations on several contrasting conditions and observe that the performance of the models range from an F-score of 0.5 to 0.8 depending on the condition. Our analysis not only reveals the inherent challenges of the task, but also provides useful directions to improve the models.

1 Introduction

In recent years, hospitals and clinics across the United States have been coaxed and cajoled into adopting Electronic Health Records through public policies and insurance requirements. This has led to the unforeseen side-effect of placing a disproportionate burden of documentation on physicians, causing burnouts among them (Wachter and Goldsmith, 2018; Xu, 2018). One study found that full-time primary care physicians spent about 4.5 hours of an 11-hour workday interacting with the clinical documentation systems, and yet were still unable to finish their documentations and had to spend an additional 1.4 hours after normal clinical hours (Arndt et al., 2017).

Speech and natural language processing are now sufficiently mature that there has been considerable interest, both in academia and industry, to investigate how these technologies can be exploited to simplify the task of documentation, and to allow physicians to dedicate more time to patients. While domain-specific ASR systems that allow doctors to dictate notes have been around for a while, recent work (Patel et al., 2018; Finley et al., 2018a,b) has begun to address more challenging tasks, such as extracting relevant information directly from doctor-patient conversations.

In this work, we investigated the task of inferring symptoms mentioned in clinical conversations, along with whether patients have experienced them or not. Our contributions include: (i) defining the task, including the annotation scheme for labeling the clinical conversations and the evaluation metrics to measure model performance (Section 3); (ii) two novel deep learning models to solve this task (Section 4); (iii) comprehensive empirical evaluations in different contrasting conditions (Section 5), and (iv) analysis of the performance of the models that provides meaningful insights for further improvements (Section 6).

2 Related Work

On the topic of information extraction from medical text, one of the earliest public-domain task is the *i2b2 challenge*, defined on a small corpus of written discharge summaries that consists of 394 reports for training, 477 for test, and 877 for evaluation (Uzuner et al., 2011). Given the small amount of training data, not surprisingly, a dispro-

[*] All the authors contributed equally.

portionately large number of teams fielded rule-based systems. CRF-based systems however did better even with the limited amount of training data. Being a written domain task, they benefited from section headings and other cues that are unavailable in doctor-patient conversations. For a wider survey of extracting clinical information from written clinical documents, see (Liu et al., 2012).

There are very few publications on processing clinical conversations. One noteworthy recent work extracts entities using a multi-stage approach (Finley et al., 2018a). They use two-level hierarchical model, modeling word sequences and sentence sequences, to classify sentences into the sections in a clinical note they belong to. The extracted sentences are then processed using a variety of heuristics such as partial string matching with an ontology, regular expressions, and other task-specific heuristics. One would imagine sentences taken out of context of a dialog are prone to misinterpretation and they do not elaborate on how that is overcome. Moreover, their system cannot be optimized end-to-end.

Other related work includes normalizing the terms and mapping them to external databases such as Unified Medical Language System (UMLS) and specific sub-tasks such as negation detection, which are outside the scope of this work (Happe et al., 2003; Nvol et al., 2014; Lowe and Huang, 2007).

3 The Symptom Extraction Task

We begin the description of our task by introducing the corpus, the annotation paradigm, and the evaluation metrics.

3.1 Corpus Description

Our unlabeled corpus consists of 90k de-identified and manually transcribed audio recordings of clinical conversations between physicians and patients, typically about 10 minutes long. A few of the conversations also contain speech from nurses, caregivers, spouses and other attendees.

The annotation guidelines were developed by a team of professional medical scribes, physicians and natural language processing experts. Two primary categories of labels were annotated: the symptoms being discussed and their status. An ontology of 186 symptoms were defined (e.g., vomiting, nausea, diarrhea), each belonging to one of 14 body systems (e.g., gastrointestinal, musculo-skeletal, cardiovascular). For each symptom, annotators were instructed to associate a status that denotes whether the patient has experienced it or not. An additional catch-all category was defined to include symptoms whose status cannot be conclusively inferred from the conversation or which are not relevant to the clinical note. Thus, status may have one of the three values: *experienced*, *not experienced*, and *other*. In an utterance, "*I have a back pain*", the underlined phrase will be assigned the tuple: (*sym:musculo-skeletal:pain, experienced*). The top three symptoms in the corpus are: musculo-skeletal pain, shortness of breath and cough.

Of the 90K encounters, we chose to focus on primary care visits. A team of 18 professional scribes was trained on the guidelines. They labeled the manual transcripts of 2,950 conversations, which were partitioned into training (1,950), development (500) and test (500) sets. The entire labeled corpus contains 5M tokens in 615K sentences with 92K labels.

To account for variation across scribes, we randomly assigned 3 scribes to label each of the conversations in the development (500) and test (500) sets. The inter-labeler agreement in terms of Cohen's kappa is 0.4 on the development set. Further analyses showed that the low score was largely due to (i) the ambiguous and informal ways that patients and doctors discuss symptoms, (ii) that human scribes often disagree on which one of closely related labels to pick, and (iii) that human scribes often disagree on the span of text to label.

3.2 Evaluation Metrics

In clinical conversations, the symptoms may be mentioned multiple times, paraphrased differently, but still may appear in the clinical notes only once. So, we chose to evaluate them at the conversation levels using two metrics.

Unweighted metric: In this metric, we account only for the unique symptoms and ignore the number of times they were mentioned. The set of events in the inferred output was compared against the set in the reference to compute the precision and recall for each conversation before averaging across all conversations.

Weighted metric: The symptoms that are mentioned more often in a conversation are likely to be more important. In this metric, each symptom is weighted by its frequency: precision is weighted by the frequency of the predictions, while recall is weighted by the frequency of the reference.

4 Models

We developed two novel neural network model architectures for this task: 1) a span-attribute model that is similar in spirit to a tagging model but works well on our large label space, and 2) a sequence-to-sequence (Seq2Seq) model (Sutskever et al., 2014; Cho et al., 2014) that is designed to infer symptoms that are described informally across a few conversation turns.

4.1 Span-Attribute Tagging (SA-T) Model

A common solution for this task is a tagging model, where the word sequences are represented by word and/or character embeddings and fed into a sequence of layers consisting of a bidirectional layer, a softmax layer and a conditional random field (CRF) to predict the BIO-style tags (Collobert et al., 2011; Huang et al., 2015; Ma and Hovy, 2016; Chiu and Nichols, 2016; Lample et al., 2016; Peters et al., 2017; Yang et al., 2017; Changpinyo et al., 2018). However, in our task, the tags need to identify not only the symptom names associated with the words but also the status. This can be accomplished in a tagging model using a label space that is the Cartesian product of both the symptom names and their status. Unfortunately, this Cartesian space turns out to quite large in our task (186 x 3). Tagging models perform well when the set of tags is reasonably small (e.g., named entity recognition and part of speech tagging), but not so well when the set of tags is large. Moreover, in our case, given the limited corpus size, modeling the cross-product space leads to data sparsity.

For tackling this challenge of data sparsity, we reformulate the problem from a novel hierarchical perspective. Unlike the conventional tagging model, where at each input token the model has to pick the best candidate label from the full label space, we break this into two stages. We first identify the span of interest using a generic tag set with a very small label set of just three elements, {*sym_B, sym_I, O*}. This simplifies the computational cost of inferring over sequence, which allows us to employ the CRF layer. Moreover, it alleviates the data sparsity problem by pooling all the labels to identify all spans of interest. In the second stage, we predict the attributes associated with the span using contextual features of arbitrary complexity without encumbering the inference over the entire sequence. In addition, since our label space can be partitioned easily, we use two separate predictors, one for symptom name and one for status. These two stages are trained jointly in an end-to-end fashion using multi-task learning, as described later.

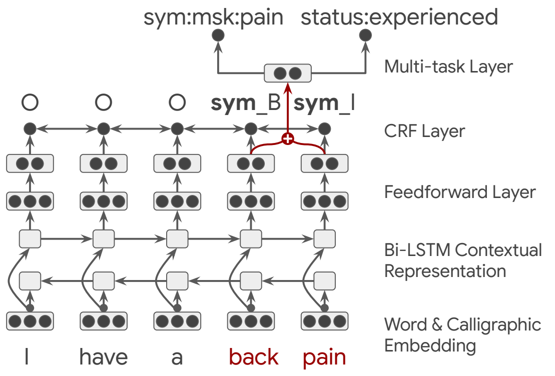

Figure 1: The architecture of Span-Attribute Tagging (SA-T) Model, illustrating the span extraction layer followed by the attribute tagging layer.

Figure 1 illustrates this hierarchical perspective for our task. The first stage, which is akin to a conventional tagging model, identifies the span of interest – *back pain* – at the output of the CRF layer. The second stage utilizes the latent representation from the span and employs two separate predictors to classify the symptom name as *sym:msk:pain* and the status as *experienced*. In principle, these predictors can be more complex than a simple soft-max that we have used. We refer to this architecture as the Span-Attribute Tagging (SA-T) model. The two stages of the model are described in more details below.

Span Extraction Layer As mentioned before, this layer employs a conventional tagging model whose output is constrained to be just three elements of $\mathcal{E} = \{sym\_B, sym\_I, O\}$. The model is briefly described as follows.

Let $\mathbf{x}$ be the embedding vector sequence corresponding to the input word sequence. From this sequence, we compute a sequence of latent contextual representations using a bidirectional LSTM, $\mathbf{h}' = [\vec{\mathbf{h}}(\mathbf{x}|\vec{\Theta}_{LSTM}), \overleftarrow{\mathbf{h}}(\mathbf{x}|\overleftarrow{\Theta}_{LSTM})]$. This latent contextual sequence is fed into a two-layer fully connected feed-forward network to obtain a final sequence of latent representation $\mathbf{h}'' = MLP(\mathbf{h}'|\Theta_{FF})$. Given this feature representation

$\mathbf{h} = (\boldsymbol{h}''_{\mathbf{1}}, \cdots, \boldsymbol{h}''_{\boldsymbol{N}})$ and the target tag sequence $\mathbf{y}^e = (y_i \,|\, i = 1, \ldots, N;\ y_i \in \mathcal{E})$, the parameters of the model are learned by minimizing the negative log-likelihood $-\log P(\mathbf{y}^e|\mathbf{h})$. This is computed in terms of a compatibility function defined over any label sequence $\mathbf{y}$ and $\mathbf{h}$.

$$S(\mathbf{y}, \mathbf{h}) = \sum_{i=0}^{N} \mathbf{A}_{y_i, y_{i+1}} + \sum_{i=0}^{N} P(\mathbf{h}_i, \mathbf{y}_i) \quad (1)$$

Of the two components, the first one estimates the probability of the label sequence in terms of the sum of first order Markov transition of the label sequence $\mathbf{y}$, computed from a learned transition matrix A whose dimensions are $|\mathcal{E}| \times |\mathcal{E}|$. The second component estimates the joint probability of the latent vector $\mathbf{h}_i$ and the corresponding label embedding $\mathbf{y}_i$, specifically, in terms of similarity measure $\mathbf{h}_i^\top \mathbf{y}_i$.

Using the compatibility function, the loss for the task of recognizing the spans is estimated as $-S(\mathbf{y}^e, \mathbf{h}) + \log \sum_{\mathbf{y}'} \exp\left(S(\mathbf{y}', \mathbf{h})\right)$, where $\mathbf{y}'$ is any other possible sequence of labels. During training, $\log P(\mathbf{y}^e|\mathbf{h})$ is estimated using forward-backward algorithm, and during inference, the most probable sequence $\mathbf{y}^* = \arg\max_{\mathbf{y}'} P(\mathbf{y}'|\mathbf{h})$ is computed using the Viterbi algorithm.

Attribute Tagging Layer Given the span, as mentioned before, we can potentially use a richer representation of the context to predict attributes than otherwise possible. A contextual representation is computed from the starting index i and ending index j of each span using a pooling function $\mathrm{Aggregate}(\cdot)$.

$$\boldsymbol{h}^s_{ij} = \mathrm{Aggregate}(\boldsymbol{h}_k | \boldsymbol{h}_k \in \mathbf{h}, i \leq k < j) \quad (2)$$

The pooling function can be implemented as simple as mean or sum, or as the hidden state of another encoder like BiLSTM, CNN or self-attention (Vaswani et al., 2017). Given the span representation $\boldsymbol{h}^s_{ij}$, we model the joint distribution of the symptom name and status as $P(y^{sx}, y^{st}|\boldsymbol{h}^s_{ij}) = P(y^{sx}|\boldsymbol{h}^s_{ij})P(y^{st}|\boldsymbol{h}^s_{ij})$ with the assumption that they are independent. Then, the distribution over the symptom name for each span is a multinomial distribution $P(y^{sx} = k|\boldsymbol{h}^s_{ij}) = \mathrm{Softmax}(\boldsymbol{h}^s_{ij}|\Theta^{sx})_k$. Similarly, we can formulate the distribution over the symptom status as $P(y^{st} = m|\boldsymbol{h}^s_{ij}) = \mathrm{Softmax}(\boldsymbol{h}^s_{ij}|\Theta^{st})_m$. Both Θ^{sx} and Θ^{st} are model parameters. Finally, we can train the model end-to-end by minimizing the following loss function for each conversation:

$$\ell(\mathbf{y}^e, \{(y^{sx}, y^{st})\}|\mathbf{h}) = -\alpha \log P(\mathbf{y}^e|\mathbf{h}) + \sum_{\{(y^{sx}, y^{st})\}} -\log P(y^{sx}|\mathbf{h}) - \log P(y^{st}|\mathbf{h}), \quad (3)$$

where $\{(y^{sx}, y^{st})\}$ is the set of symptom names and associated status in a conversation, and α is the relative weight of the loss of the span extraction task and the attribute prediction task.

During training, we are simultaneously attempting to detect the location of tags as well as classify the tags. Initially, our model for locating the tags is unlikely to be reliable, so we adopt a curriculum learning paradigm. Specifically, we provide the classification stage the reference location of the tag from the training data with probability p, and the inferred location of the tag with probability $1 - p$. We start the joint multi-task training by setting this probability to 1 and decrease it as training progresses (Bengio et al., 2015).

Remarks Although the SA-T model was developed to infer symptoms and status in the clinical domain, the formulation is general and can be applied to any domain. The model breaks up the task into identifying spans of interests and then classifying the span with richer contextual representations. The first stage alleviates data sparsity by pooling all spans of interest. When the label space naturally partitions into separate categories, the second stage can be broken up further into separate prediction tasks and reduces data splitting.

4.2 Sequence-to-sequence (Seq2Seq) Model

As shown in Table 1, symptoms are sometimes not stated explicitly, but rather explained or described in informal language over several conversation turns. There may not even be a symptom entity that is explicitly mentioned; instead, the physician, as well as any symptom extraction model, must infer it from a description. To better capture symptoms that are not referred to by name, we explore an alternative formulation of the problem. In this formulation, the input to the model is a chunk of the conversation, consisting of multiple consecutive turns from the doctor-patient conversation, and the output is a list of symptoms mentioned as well as their statuses. The key difference between this formulation and the span-attribute tagging formulation is that the symptom entity is not assigned to a word or phrase in the input text.

Transcript	Symptoms + Status
DR: Any issues with your eyes? **PT**: Well sort of **DR**: Is your vision ok? **PT**: Yeah, but the right one hurts	**Eye pain**: experienced **Vision loss**: not experienced
DR: How is your bladder? **PT**: I have to go, all the time **DR**: At night? **PT**: No, just during the day	**Frequent urination**: experienced **Nocturia**: not experienced

Table 1: Two illustrative examples where symptoms and their status are not described explicitly but need to be inferred from the context spanning multiple turns.

In this formulation, each input example consists of a segment of transcript, represented as a sequence of tokens $\mathbf{x} = (x_1, ..., x_m)$, and a list of symptoms and their corresponding status $\mathbf{y} = (y_1, ..., y_n)$. Hence, it is well-suited to the sequence-to-sequence (Seq2Seq) class of models (Sutskever et al., 2014; Cho et al., 2014) which has been successful across a variety of language understanding tasks, including conversation modeling (Vinyals and Le, 2015), abstractive summarization (Nallapati et al., 2016), and question answering (Seo et al., 2017). Following the standard Seq2Seq setup, our model is composed of two recurrent neural networks (RNNs), an encoder and a decoder. First, the *encoder* consumes $\mathbf{x}$ one token at a time, producing an encoding, $\boldsymbol{h}(x_i)$, for each token x_i. Then the *decoder* estimates an output distribution over sequences of symptoms and their status $\mathbf{y}$, conditional on the encodings. An attention mechanism (Bahdanau et al., 2015) allows the decoder to combine information from the encoded sequences differently at each decoding step.

The Seq2Seq model is trained using a cross-entropy criterion to maximize $P(\mathbf{y}|\mathbf{x})$ – the likelihood of reference symptoms and their status given the conversation transcripts. At inference time, the most likely sequence of symptoms and their status is decoded one token a time using beam search. One challenge for Seq2Seq models is handling very long inputs (Sutskever et al., 2014). Therefore, unlike the span-attribute tagging model where each input example may be a full transcript, we use transcript segments consisting of k consecutive turns. In practice we found a value of $k = 5$ to work well. A value of k that is too small won't be enough to resolve symptoms like those in Table 1, while a value of k that is too large may degrade quality and make our model harder to train. At inference time, we use a sliding window of size k across the full conversation, and then aggregate the predictions from those windows.

4.3 Encoder Pre-training

While the span-attribute tagging and Seq2Seq models have different output layers, they use a common input encoder architecture. At any given input time, the conversation up to that time is represented by the hidden state of the encoder, which is used for making output predictions. We investigated two variations of the encoder.

First, we compare the LSTM encoder with the Transformer encoder (Vaswani et al., 2017). The key difference between them is that the LSTM relies on latent variables to propagate state information while Transformer relies solely on an attention mechanism. In a machine translation benchmark, the Transformer has been shown to outperform the LSTM encoder (Vaswani et al., 2017), and a hybrid model, consisting of a Transformer encoder and an LSTM decoder, performed even better (Chen et al., 2018). We therefore compare the hybrid model, with the LSTM-only encoder-decoder model on our task.

Second, we use a *pre-training* technique to leverage unlabeled data and improve the feature representation learned by the encoder (Kannan et al., 2018). Given a short snippet of conversation, the model is tasked with predicting the next turn, similar to Skip Thought (Kiros et al., 2015). Since this task requires no labeling, the model can be trained on the full corpus of 90K conversations. The resulting encoder is plugged into our model for the symptom prediction task, and the full model is trained on the subset that is labeled. The pre-training can be performed for both the LSTM and Transformer encoders, as well as for both the Seq2Seq and the span-attribute tagging models. We did not experiment with alternative pre-training loss such as BERT (Devlin et al., 2018).

5 Empirical Evaluations

Before creating dedicated models for this task, we investigated general purpose named-entity annotation tools akin to (Momchev, 2010; Nothman et al., 2008). While a few of these tools can annotate symptom entities with some accuracy, they have no mechanism to infer the symptom status, which is required for clinical documentation.

In all the experiments described below, our models were trained and evaluated on the corpus described in Section 3.1 using the metrics defined in Section 3.2. Since our ontology differs from the public domain *i2b2* task, we could not evaluate our models on that task.

For a robust estimate of the model performance, the model outputs were evaluated against a “voted” reference created using the labels from three independent scribes. This is the case for all the results reported in the experiments below, unless otherwise specified. While our application requires jointly inferring both the symptom and status (Sx + Status), for a better understanding of the model behavior we have also included the performance on inferring just the symptom names (Sx). These are reported in separate columns in the tables below.

5.1 Hyperparameters

The hyperparameters of the Span-Attribute tagging (SA-T) and the Seq2Seq models were picked to maximize the performance on the development set. The models were trained using the Adam optimizer (Kingma and Ba, 2015) and the selected parameters are reported in Table 2.

Parameter	SA-T	Seq2Seq	Range
Word emb	256	256	[128 – 512]
LSTM Cell	1024	512	[256 – 1024]
Enc/dec layers	1	1	[1 – 3]
Dropout	0.4	0.0	[0.0 – 0.5]
L2	1e-4	1e-4	[1e-5 – 1e-2]
Std of VN	1e-3	0.2	[1e-4 – 0.2]
α of SA-T	0.01	n/a	[1e-4 – 0.1]
Learning rate	1e-2	3e-3	[1e-4 – 1e-1]

Table 2: The range over which hyperparameters were tuned and the optimal choice for each model.

5.2 Different Encoders and Pre-training

To select the encoder, first we evaluate the impact of pre-training on the LSTM encoder, using the Seq2Seq model. The results are reported in Table 3. The results show that pre-training of the LSTM encoder consistently improves performance of the Seq2Seq model across all metrics.

Next, the Transformer encoder was compared against the LSTM encoder, using pre-training in both cases. Based on the performance on the development set, the best encoder was chosen which consists of two layers, each with 1024 hidden dimension and 16 attention heads. The results in Table 4 show that the LSTM-encoder outperforms the Transformer-encoder consistently in this task, when both are pre-trained. Therefore, for the rest of the experiments, we only report results using the LSTM-encoder.

Pretrained	Sx	Sx + Status
	Unweighted F1(Precision, Recall)	
No	**0.69** (0.66, 0.73)	**0.54** (0.49, 0.60)
Yes	**0.70** (0.66, 0.75)	**0.55** (0.49, 0.62)
	Weighted F1(Precision, Recall)	
No	**0.77** (0.76, 0.78)	**0.63** (0.60, 0.65)
Yes	**0.79** (0.77, 0.80)	**0.64** (0.61, 0.68)

Table 3: The comparison of Seq2Seq model performance when the LSTM encoder is intialized randomly and when the encoder is pre-trained on the entire corpus including the unlabelled data.

Encoder	Sx	Sx + Status
	Unweighted F1(Precision, Recall)	
Xformer	**0.67** (0.66, 0.67)	**0.51** (0.48, 0.54)
LSTM	**0.70** (0.66, 0.75)	**0.55** (0.49, 0.62)
	Weighted F1(Precision, Recall)	
Xformer	**0.76** (0.79, 0.74)	**0.61** (0.62, 0.61)
LSTM	**0.79** (0.77, 0.80)	**0.64** (0.61, 0.68)

Table 4: The comparison of Seq2Seq model performance using Transformer (Xformer) and LSTM encoders. Both encoders were pre-trained.

5.3 Manual Transcript Evaluation

Next, we evaluate and compare the performance of the models when they are trained and tested on the manual transcripts. For comparison, we include a standard tagging baseline consisting of a bidirectional LSTM-encoder (pre-trained as described in Section 4.3), followed by two feed-forward layers and a softmax layer. The targets consisted of the cross product space of 186 symptom names and 3 status values. The model was trained using cross-entropy loss. Due to the large cross product label space, the CRF loss is infeasible in this setting.

From the results reported in Table 5, we see that the span-attribute tagging model performs as well as the Seq2Seq model. This is surprising since it is designed to not only predict the symptom name and status, but also to locate the words associated with them, a more demanding task. Another noteworthy difference between the two models is that the tagging model consistently trades off lower recall for higher precision, compared to the Seq2Seq model. The Mann-Whitney rank test indicates that improvements of both the models over the baseline are statistically significant under both metrics. In

Model	Sx	Sx + Status
Unweighted F1(Precision, Recall)		
Baseline	**0.68** (0.73, 0.63)	**0.50** (0.54, 0.47)
SA-T	**0.71** (0.73, 0.69)	**0.58** (0.58, 0.58)
Seq2Seq	**0.70** (0.66, 0.75)	**0.55** (0.49, 0.62)
Weighted F1(Precision, Recall)		
Baseline	**0.73** (0.78, 0.69)	**0.57** (0.61, 0.53)
SA-T	**0.77** (0.80, 0.74)	**0.65** (0.66, 0.63)
Seq2Seq	**0.79** (0.77, 0.80)	**0.64** (0.61, 0.68)

Table 5: The comparison of performance on manual transcripts between the baseline, the SA-T and the Seq2Seq models.

general, a gain of about 0.02 or more in F1-score was found to be statistically significant in our experiments on this task.

Knowing that the quality of the reference impacts the measured performance, we compared the model output to two versions of references in addition to the "voted" reference. In one version, we used a single reference for each conversation from a randomly chosen scribe. In another version, the model was given credit when the output matches "any" of the three scribes. This was motivated by the observation during adjudication that the symptom names may be annotated in more than one way, as illustrated in the example in Table 6.

PT: I found the exercises very difficult.
DR: Was it hurting you?
PT: Yeah, a lot.

Table 6: An illustrative example to show how symptom (*hurting*) may be assigned either symptom names – *sym:musculo_skeletal:pain* or *sym:constitutional:pain*, which are both valid given the context.

The model outputs were compared against the above mentioned variants of the reference and the results are reported in Table 7. The measured gap in performance between single reference and "voted" reference is small. The "voted" version corrects the reference, when one of the three scribes misses the annotation. However, when two scribes pick different valid labels and the third misses them, the "voted" reference is not better than the single reference. In such instances, allowing a model to match "any" of the references would be a reasonable solution. This may explain why the performance in that case is substantially better than the single or "voted" reference.

	Ref.	Sx	Sx + Status
Unweighted F1(Precision,Recall)			
SA-T	Single	**0.70** (0.72, 0.69)	**0.56** (0.56, 0.57)
	Voted	**0.71** (0.73, 0.69)	**0.58** (0.58, 0.58)
	Any	**0.81** (0.84, 0.78)	**0.69** (0.71, 0.67)
Seq2Seq	Single	**0.68** (0.62, 0.76)	**0.53** (0.45, 0.63)
	Voted	**0.70** (0.66, 0.75)	**0.55** (0.49, 0.62)
	Any	**0.81** (0.77, 0.84)	**0.67** (0.62, 0.73)
Weighted F1(Precision, Recall)			
SA-T	Single	**0.76** (0.79, 0.74)	**0.63** (0.64, 0.62)
	Voted	**0.77** (0.80, 0.74)	**0.65** (0.66, 0.63)
	Any	**0.86** (0.89, 0.83)	**0.75** (0.77, 0.73)
Seq2Seq	Single	**0.77** (0.73, 0.80)	**0.62** (0.57, 0.68)
	Voted	**0.79** (0.77, 0.80)	**0.64** (0.61, 0.68)
	Any	**0.87** (0.86, 0.89)	**0.75** (0.72, 0.78)

Table 7: The comparison of model performance on manual transcripts when the performance was evaluated against *Single*, *Voted* and *Any* reference labels.

5.4 ASR vs. Manual Transcript Evaluation

In clinical applications, manual transcripts will be unavailable and the model needs to infer symptom and status on transcripts obtained from an automatic speech recognition (ASR) system. We investigated the impact on performance when the test data is switched from manual to the corresponding ASR transcripts. Such a switch is expected to degrade the performance of models trained on manual transcripts and often this degradation can be alleviated by training the model on ASR transcripts. So, we measured performance using models trained on different combinations of manual and ASR transcripts.

Recall, the symptom, as described in Section 3.1, were annotated on manual transcripts. These annotations were automatically transferred to the ASR transcripts by aligning the words in both transcripts for the same speaker turns and mapping the labels from manual transcripts to the corresponding words in the ASR transcripts. The word error rate of the ASR transcripts is about 20% (Chiu et al., 2018). In the alignment process, a fraction of the labels (9.1%) failed to alignment properly and were discarded.

The results, reported in Table 8 with "voted" reference, show that the performance of the models trained on manual transcripts (*Manual Train*) degraded when tested on ASR transcripts (*ASR Test*), for both models, as expected. But, surprisingly, training models on ASR transcripts (*ASR*

Model	Type	Manual Test	ASR Test
	Unweighted F1(Sx, Sx+Status)		
SA-T	Manual Train	0.71, 0.58	**0.67**, **0.52**
	ASR Train	0.68, 0.55	0.67, 0.52
	Combined	**0.72**, **0.59**	0.66, 0.53
Seq2Seq	Manual Train	0.70, 0.55	0.65, 0.50
	ASR Train	0.67, 0.50	0.62, 0.47
	Combined	0.69, 0.54	0.64, 0.49
	Weighted F1(Sx, Sx+Status)		
SA-T	Manual Train	0.77, 0.65	0.72, 0.58
	ASR Train	0.75, 0.62	0.72, 0.58
	Combined	0.78, 0.65	0.71, 0.58
Seq2Seq	Manual Train	**0.79**, **0.64**	**0.75**, **0.59**
	ASR Train	0.76, 0.61	0.72, 0.57
	Combined	0.79, 0.64	0.74, 0.59

Table 8: The comparison of model performances when trained on manual (Manual Train), ASR (ASR Train), and their combined (Combined) transcripts and evaluated on manual (Manual Test) and ASR (ASR Test) transcripts. The best performance is shown in bold.

Model	Sx + Status	Sx System + Status
	Unweighted F1(Precision, Recall)	
Baseline	n/a	**0.60** (0.67, 0.54)
SA-T	**0.58** (0.57, 0.58)	**0.69** (0.70, 0.69)
Seq2Seq	**0.55** (0.49, 0.62)	**0.67** (0.62, 0.73)
	Weighted F1(Precision, Recall)	
Baseline	n/a	**0.68** (0.75, 0.62)
SA-T	**0.65** (0.66, 0.63)	**0.77** (0.79, 0.76)
Seq2Seq	**0.64** (0.61, 0.68)	**0.78** (0.76, 0.81)

Table 9: The comparison of model performances when the symptom names (Sx) are collapsed to their respective body system (Sx System) categories.

Model	Sx	Sx + Status
	Unweighted F1(Precision, Recall)	
Human	**0.84** (0.86, 0.82)	**0.78** (0.80, 0.76)
SA-T	**0.71** (0.73, 0.69)	**0.58** (0.58, 0.57)
Seq2Seq	**0.70** (0.66, 0.75)	**0.55** (0.49, 0.62)
	Weighted F1(Precision, Recall)	
Human	**0.86** (0.88, 0.85)	**0.81** (0.82, 0.79)
SA-T	**0.77** (0.80, 0.74)	**0.65** (0.66, 0.63)
Seq2Seq	**0.79** (0.77, 0.80)	**0.64** (0.61, 0.68)

Table 10: The comparison of performance of models and single scribes against the "voted" reference.

Train) or folding the ASR transcripts into the manual training data (*Combined*) did not improve the performance much. This maybe due to the fact that our performance metrics are evaluated at the conversation level and there is redundancy in clinical conversations, where the same symptom may be mentioned multiple times during the course of the conversation and each time in a different way.

5.5 Symptom Names vs. Body Systems

One way to understand the confusion between symptom names is to measure the performance after projecting the inferred symptom names (186 types) to their corresponding body systems (14 types). For example, *sym:musculo-skeletal:pain* and *sym:musculo-skeletal:swelling* were collapsed to *sym:musculo-skeletal.*

As a baseline, we trained an LSTM tagger with a CRF output layer to predict targets consisting of the simple Cartesian product of symptom body systems and their status. The performance of the baseline system and our models were evaluated on manual transcripts. Our models were trained to predict the symptom name and the predictions were projected to the system level. The results are reported in Table 9.

When the symptom names are collapsed into broader body systems, the performance improves as expected. The gain in performance is surprisingly large at about 0.14 F1-score. This suggests that a large fraction of confusion comes from names in the same body system. The baseline model has much lower precision and recall compared to our proposed models, even though it was trained on the body system labels directly, once again, demonstrating that the cross-product space is too sparse to be learned properly.

6 Analysis

In this section, we conduct detailed comparisons among human scribes and our models.

6.1 Human Performance

To understand the inherent difficulty of this task, we estimated the human performance on this task by comparing each scribe against the reference generated from the "voted" results of the three scribes. Even though this estimate is inflated, because each scribes' annotation was counted towards the voted reference, it is a useful approximation. The results in Table 10 show two clear trends. First, even humans have difficulty identifying symptoms consistently. For example, "constitutional pain" (non-specific) and "musculo-skeletal pain" were top confusions for our models as well as humans. Second, when status is considered, humans have less trouble inferring it from the context than our models, losing only 0.05 on F1 (weighted), while our models dropped about 0.14. Improving status classification remains one of our future work.

6.2 Attention Weights

Next, we inspected the Seq2Seq model's attention weights to see whether the evidence is scattered across words and turns in the dialog. Indeed, through manual inspection, we found this to be true qualitatively, as illustrated in Table 11. In this example, the symptom "sym:const:difficulty sleeping" is not mentioned directly but is implied from the evidence scattered in the context. Future work could use these weights to further investigate errors.

DR: How is your sleep?
PT: Well, I have been waking up a lot.
DR: How often would you say?
PT: Several times a night.
DR: That is a lot of waking up!

Table 11: Example of attention from Seq2Seq model, where words with attention weight of 0.05 or higher are underlined.

6.3 Error Analysis

Grouping false negatives by their symptom name, we observed that both models struggled with the symptoms – pain, malaise, fatigue, difficult sleeping, weight loss/gain, and frequent urination. As illustrated in Table 12, these symptoms were often communicated through back-and-forth with the doctor and therefore may have required combining evidence from multiple turns, making the inference more difficult.

Muscoloskeletal pain
DR: Does it hurt when you go like this?
PT: No, that shoulder is fine.
DR: So this side hurts, but that side, if you reach, there's no pain?
PT: Yeah, really only this one has been sore.

Weight loss/gain
DR: Okay. So when you took these, it went up?
PT: Well it was high, then I lost a few pounds. Then just, it's been really stressful, I've slipped.
DR: So it went back up?
PT: Yeah, it's been up and down.

Table 12: Examples of evidence spreading across multiple turns.

7 Conclusions

This paper describes a novel information extraction task, that of extracting the symptoms mentioned in clinical conversations along with their status. We describe our corpus, the annotation paradigm, and tailored evaluation metrics. We proposed a novel span-attribute tagging (SA-T) model and a variant of sequence-to-sequence model to solve the problem. The SA-T model breaks up the task into identifying spans of interests and then classifying the span with richer contextual representations. The first stage alleviates data sparsity by pooling all spans of interest. When the label space naturally partitions into separate categories, the second stage can be broken up further into separate prediction tasks and reduces data splitting. Although the SA-T model was developed to infer symptoms and status in the clinical domain, the formulation is general and can be applied to any domain. As an alternative, our Seq2Seq model is designed to infer symptom labels when the evidence is scattered across multiple turns in a dialog and is not easily associated with a specific word span. The performance of our models is significantly better than baseline systems and range from an F-score of 0.5 to 0.8 depending on the condition. When the models are trained on manual transcripts and applied on ASR transcripts, the performance degrades considerably compared to applying them on manual transcripts. Training the model on ASR transcripts or on both ASR and manual transcripts does not help bridge the performance gap. Our analysis show that the SA-T model has higher precision while Seq2Seq model has higher recall, thus the two models compliment each other. We plan to investigate the impact of combining the two models.

Acknowledgments

This work would not have been possible without the help of a number of colleagues, including Gang Li, Mingqiu Wang, Laurent El Shafey, Hagen Soltau, Patrick Nguyen, Nina Gonzales, Diana Jaunzeikare, Philip Chung, Ashley Robson Domin, Lauren Keyes, Alvin Rajkomar, Justin Stuart Paul, Katherine Chou, Chris Co, Claire Cui, and Kyle Scholz.

References

Brian G. Arndt, John W. Beasley, Michelle D. Watkinson, Jonathan L. Temte, Wen-Jan Tuan, Christine A. Sinsky, and Valerie J. Gilchrist. 2017. Tethered to the EHR: primary care physician workload assessment using EHR event log data and time-motion observations. *Annals of Family Medicine*, 15(5):419–26.

D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR)*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1171–1179, Cambridge, MA, USA. MIT Press.

Soravit Changpinyo, Hexiang Hu, and Fei Sha. 2018. Multi-task learning for sequence tagging: An empirical study. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2965–2977. Association for Computational Linguistics.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *ACL*.

Chung-Cheng Chiu, Anshuman Tripathi, Katherine Chou, Chris Co, Navdeep Jaitly, Diana Jaunzeikare, Anjuli Kannan, Patrick Nguyen, Hasim Sak, Ananth Sankar, Justin Tansuwan, Nathan Wan, Yonghui Wu, and Xuedong Zhang. 2018. Speech recognition for medical conversations. In *Interspeech*, pages 2972–2976. ISCA.

Jason Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

K. Cho, B. van Merriënboer, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proc. EMNLP*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *https://arxiv.org/abs/1810.04805*.

Greg P. Finley, Erik Edwards, Amanda Robinson, Najmeh Sadoughi, Mark Miller, David Suendermann-Oeft, and Nico Axtmann Michael Brenndoerfer. 2018a. An automated medical scribe for documenting clinical encounters. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Gregory Finley, Wael Salloum, Najmeh Sadoughi, Erik Edwards, Amanda Robinson, Nico Axtmann, Michael Brenndoerfer, Mark Miller, and David Suendermann-Oeft. 2018b. From dictations to clinical reports using machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 121–128. Association for Computational Linguistics.

André Happe, Bruno Pouliquen, Anita Burgun, Marc Cuggia, and Pierre Le Beux. 2003. Automatic concept extraction from spoken medical reports. *I. J. Medical Informatics*, 70(2-3):255–263.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *https://arxiv.org/abs/1508.01991*.

Anjuli Kannan, Kai Chen, Diana Jaunzeikare, and Alvin Rajkomar. 2018. Semi-supervised learning for information extraction from dialogue. *Interspeech*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28*, pages 3294–3302.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Feifan Liu, Chunhua Weng, and Hong Yu. 2012. *Natural Language Processing, Electronic Health Records, and Clinical Research*, pages 293–310. Springer Science & Business Media.

Henry J. Lowe and Yang Huang. 2007. A novel hybrid approach to automated negation detection in clinical radiology reports. *Journal of the American Medical Informatics Association*, 14(3):304–311.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074. Association for Computational Linguistics.

N. Momchev. 2010. Annotating web documents with Wikipedia entities. Master's thesis, Sofia University.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *ACL*.

Joel Nothman, James R Curran, and Tara Murphy. 2008. Transforming wikipedia into named entity training data. In *Proceedings of the Australasian Language Technology Association Workshop*.

Aurlie Nvol, Cyril Grouin, Jeremy Leixa, Sophie Rosset, and Pierre Zweigenbaum. 2014. The quaero french medical corpus: A ressource for medical entity recognition and normalization. In *In Proc BioTextM, Reykjavik*.

Pinal Patel, Disha Davey, Vishal Panchal, and Parth Pathak. 2018. Annotation of a large clinical entity corpus. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2033–2042. Association for Computational Linguistics.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

I. Sutskever, O. Vinyals, and Q. Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems (NIPS)*.

Ozlem Uzuner, Brett R South, Shuying Shen, and Scott L. DuVall. 2011. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of American Medical Informatics Association*, 18(5):552–6.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

O. Vinyals and Q. Le. 2015. A neural conversation model. In *ICML Deep Learning Workshop*.

Robert Wachter and Jeff Goldsmith. 2018. To combat physician burnout and improve care, fix the electronic health record. *Harvard Business Review*.

Rena Xu. 2018. The burnout crisis in american medicine. *The Atlantic*.

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *ICLR)*.