Joint Learning with Pre-trained Transformer on Named Entity Recognition and Relation Extraction Tasks for Clinical Analytics

Miao Chen
Covance
miao.chen
@covance.com

Ganhui Lan
Janssen
ganhuilan
@gmail.com

Fang Du
Covance
fangdu64
@gmail.com

Victor Lobanov
Covance
victor.lobanov
@covance.com

Abstract

In drug development, protocols define how clinical trials are conducted, and are therefore of paramount importance. They contain key patient-, investigator-, medication-, and study-related information, often elaborated in different sections in the protocol texts. Granular-level parsing on large quantity of existing protocols can accelerate clinical trial design and provide actionable insights into trial optimization. Here, we report our progresses in using deep learning NLP algorithms to enable automated protocol analytics. In particular, we combined a pre-trained BERT transformer model with joint-learning strategies to simultaneously identify clinically relevant entities (i.e. Named Entity Recognition) and extract the syntactic relations between these entities (i.e. Relation Extraction) from the eligibility criteria section in protocol texts. When comparing to standalone NER and RE models, our joint-learning strategy can effectively improve the performance of RE task while retaining similarly high NER performance, likely due to the synergy of optimizing toward both tasks' objectives via shared parameters. The derived NLP model provides an end-to-end solution to convert unstructured protocol texts into structured data source, which will be embedded into a comprehensive clinical analytics workflow for downstream trial design missions such like patient population extraction, patient enrollment rate estimation, and protocol amendment prediction.

1 Introduction

Clinical trial protocols, often called "study protocols" or just "protocols", are the foundational documents that specify the detailed plans of conducting clinical trials to validate the safety and/or efficacy of drugs. They contain key information about the targeted disease indications, the eligible patients, the investigated medication, the visit schedules, and the treatment endpoints etc. Across the entire lifecycle of clinical trials starting from study design & planning to data analysis & publication, it is always critical to comprehend this information accurately and unambiguously. However, since protocols are mainly unstructured or semi-structured texts (i.e. natural languages), application of computer-aided information extraction is challenging and thus limited. Current protocol analytic practices are labour- and time-intensive, involving numerous manual resource checking and cross referencing activities. The pressing needs of reducing the costs and boosting the speed of drug development have created an industry-wide demand in developing a more efficient, effective, and scalable mechanism to process text-based protocols.

To address the above demand, we present in this paper our efforts and progresses in developing a deep learning Natural Language Processing (NLP) approach to extract clinically relevant information from protocols. In particular, we targeted two tasks: Named Entity Recognition (NER) and Relation Extraction (RE), and transferred the Bidirectional Encoder Representations from Transformers model (BERT, a pre-trained transformer NLP model) via a joint-learning strategy to extract clinically relevant entities and their syntactic relationships simultaneously by training on our in-house clinical trial protocol corpus.

In alignment with the industry's patient-centric business emphasis, we focused this work on extracting the patient eligibility information from the "Eligibility Criteria" section in the protocols, which unambiguously determines whether a patient could be included in or excluded from the clinical trial. This is particularly important because patient recruitment is an essential and currently rate-limiting step in clinical trials. Accurate parsing of this part of protocols can facilitate quick identification of eligible patients as well as other clinical analytics

Proceedings of the 3rd Clinical Natural Language Processing Workshop, pages 234–242

missions.

Clinical trial protocols are a type of professional documents with rigorous and highly domain-specific terms associated via complex yet precise relations. Like other professionally developed documents, protocols have to pass multiple quality-control checkpoints, and thus require less pre-processing (e.g. text correcting/cleaning) than many other types of documents such as social media posts before submitting to NLP models. On the other hand, the domain-specific nature of protocols requires extra attention when transferring models trained from generic or other professional domains. To elaborate, a protocol contains many clinical and medical terms (e.g. medications and diseases etc.) that are not commonly seen in other domains, but those are exactly the entities that our model needs to recognize; furthermore, it is also challenging that the entities may be connected in dramatically different ways under different domain-specific contexts. For example, in the clinical domain, the word "trial" refers to "clinical trial" that is associated with "patients", "diseases", and "medicines" etc.; whereas in the legal or even generic domains, "trial" commonly means "legal trial" that is frequently connected to "jury", "prosecutor" and "defendant" etc. Therefore, the success of the transfer-learning largely relies on maximizing domain-specific "gradients" for fine-tuning the model parameters.

This presented work is continued from our recent study on clinical protocols, in which we developed standalone BERT-based NLP models for NER and RE tasks for processing the "Eligibility Criteria" section in protocols (Chen et al., 2020).

Based on the observation that different clinically relevant entities are not equally involved in all relations, we hypothesized that by combining the NER and RE tasks in the same BERT network via a joint-learning strategy, the textures of clinical trials may become more visible for training and thus improve the performance of both tasks. As will be shown in the later sections of this paper, our results validated this hypothesis and showed that the joint-learning model can provide significant improvement over standalone models.

This improved model is being embedded into an automated pipeline that aims to accelerate the current manual process of identifying similar clinical trials from the historical protocols, and to streamline the querying process of identifying potentially eligible patients for clinical trials.

2 Related Work

NER and RE are two classic NLP tasks that have been studied separately for decades. In its earlier developments, NER, as a sequence labeling task, has mainly employed probabilistic sequence labeling techniques such as conditional random fields (CRF), maximum entropy Markov models, and hidden Markov models (Lafferty et al., 2001; McCallum et al., 2000; Bikel et al., 1998). More recently, researchers have started using deep learning family of algorithms to capture the transitions between hidden states for NER tasks, including recurrent neural networks (RNN), bidirectional long short-term memory (BiLSTM) together with CRF. Lately, pre-trained transformer models, with BERT as a prominent example, have been developed and used to represent contextual embeddings of text and gained great success in NER tasks along with other NLP tasks (Devlin et al., 2018; Lee et al., 2019).

With regard to RE, it is usually treated as a text classification between the interested entity pairs. Many classification algorithms, such as support vector machine, logistic regression, perceptron etc., have been applied to this problem (Bach and Badaskar, 2007; Jurafsky, 2000). Similar to NER, the latest developments in solving RE tasks have also employed deep learning algorithms using neural network models to emulate entity relations with components such as attention, biaffine, and bidirectional tree-structured LSTM-RNNs (Nguyen and Verspoor, 2019; Wang et al., 2019a; Miwa and Bansal, 2016). Pre-trained models were also used to provide contextualized encoding information to the neural network layers for the RE task (Lee et al., 2019; Wang et al., 2019b).

Although they can be tackled independently, NER and RE tasks are in fact synergistically connected: if we knew two entities and their types in a sentence, it would be easier to classify their relations; similarly, if we knew the relation between two phrases, then it would be less challenging to label their entity types. This has naturally motivated efforts in joint or multi-task learning for NER and RE, hoping to achieve better performances in both tasks by simultaneously training the same neural network towards combined objectives. Despite the differences in their details, the practices in NER and RE joint learning usually share a general high-level architecture: they sequentially stack together the word and sequence embedding layers, the NER

prediction layer, the NER entity embedding layer, and the relation representation/handling layers. For word and sequence contextualized embedding layers, where many network variations be present, researchers have investigated using BiLSTM, RNN, and BERT pre-trained transformers. (Bekoulis et al., 2018b; Wang et al., 2019a; Giorgi et al., 2019; Huang et al., 2019b; Katiyar and Cardie, 2017). These studies usually emphasized more on evaluating different joint models, leaving the comparison between joint and standalone models to be investigated.

Pre-trained transformer models, e.g. BERT, XL-Net, and GPT, have achieved state-of-the-art performance across a great number of benchmark NLP tasks (Devlin et al., 2018; Yang et al., 2019; Radford et al., 2018). They provide the benefits of representing bidirectional context and encoding text sequence by a series of attention layers. From the transfer learning standpoint, various NLP tasks can be treated as downstream tasks appended to the pre-trained models, and the pre-trained parameters (usually from large scale corpora in a generic domain) together with the NLP task specific parameters are fine-tuned via continued training on a relatively smaller and task-specific training data set. To enhance domain specificity, BERT has also been customized and retrained on specific domains such as the biomedical domain against relevant corpus, examples including BioBERT, ClinicalBERT, and SciBERT (Lee et al., 2019; Alsentzer et al., 2019; Beltagy et al., 2019). Also, there has been a surge in studies applying BERT in specific NLP contexts for fine-tuning tasks such as predicting hospital re-admission, extracting bacteria-biotope relations, biomedical named entity normalization, etc. (Huang et al., 2019a; Jettakul et al., 2019; Li et al., 2019).

We have previously investigated the applications of fine-tuning pre-trained BERT models on a protocol corpus for NER and RE tasks separately. Encouraged by many successful studies on pre-trained transformer and joint models, we continued to explore joint-learning strategies combined with BERT to co-train NER and RE tasks against our in-house clinical protocol corpus. We abstracted a neural architecture from two popular joint models and experimented with a number of variations (Bekoulis et al., 2018b; Giorgi et al., 2019). We believe these continued efforts not only help selecting the best-performing model for our applications, but also provide a comprehensive understanding of various transfer learning strategies' performance under a real-world setup, shedding light on developing business-oriented AI applications for the healthcare and clinical trial industry.

3 Data Set

Data. Our data is comprised of the eligibility criteria sections from 470 Covance in-house drug development study protocols (less than 2% of the total number of in-house protocols). The eligibility criteria section in a protocol explicitly and unambiguously defines the rules to include or exclude a patient, thus directly determining the patient population available for the trial. The corpus contains 30,183 criteria sentences in total. We randomly split the sentences into training and test sets with a 2:1 ratio, resulting in 20,122 sentences for training and 10,061 for test.

The sentences were manually annotated by biomedical experts. Clinically relevant entities and the associated entity relations are labelled based on our annotation guideline. We used the BIO tag format to denote the beginning, inside, and outside of the entities (Ramshaw and Marcus, 1999). We focused on 15 types of entities and 7 types of syntactic relations (as shown in Table 1 and Table 2):

Table 1: Train and test data counts for the NER task.

Entity	Train	Test
Condition	12,682	8,537
Observation	7,309	5,218
Procedure	3,406	2,234
Device	221	140
Drug	7,793	5,858
Investigational product	329	224
Event	2,430	1,625
Refractory condition	381	278
Demographics	498	381
Measurement	4,540	3,344
Temporal constraints	6,968	4,589
Qualifier/modifier	7,853	5,196
Anatomic location	427	223
Negation cue	921	615
Permission cue	1,236	869

4 Methodology

4.1 Joint Model Overview

After reviewing the previous NER and RE joint models, we established a general network architecture that includes key components for the joint learning while allowing experiment using varia-

Table 2: Train and test data counts for the RE task.

Relation	Train	Test
is negated	703	468
is permitted	1,009	673
modified by	5,715	3,810
has value	3,326	2,218
has temporal constraint	6,169	4,112
is located	215	143
specified by	3,729	2,486
total count	20,866	13,910

tions in local network designs. The main structure of the joint model is shown in Figure 1.

We used BERT pre-trained transformer as the embedding/encoding layer. The NER layer occurs after the BERT layer and uses softmax for NER classification. More specifically, it takes the BERT output vectors as its input and first passes through a fully connected layer and then the output layer where NER labels are classified using softmax function (Goodfellow et al., 2016). The NER classification loss function based on cross-entropy is:

$$loss_{NER} = \sum_{i=1}^{n} -log(\frac{e^{s_{i,l_i}}}{\sum_{j=1}^{k} e^{s_{i,c_j}}}) \quad (1)$$

where n is the total number of NER tokens, l_i is the actual NER label for the i^{th} token, k is the number of NER label classes, c_j denotes any of the NER label classes, s_{i,l_i} is the linear score for the i^{th} token belonging to its actual class l_i, and s_{i,c_j} is the linear score for the i^{th} token belonging to entity class c_j.

Following the NER layer, we appended an NER label embedding layer, which is concatenated with the outputs from the previous BERT layer to serve as the input for the subsequent RE task. Because an entity could be paired with other entities before or after it in a sentence, it should be mapped differently in these 2 cases. In our model, the entity vectors are processed in the relation pair handling layer, by 1) mapping them using a fully connected layer to head vectors for representing entities as heads in a pair, and 2) mapping them using a parallel fully connected layer to tail vectors for representing tails in a pair. Then an entity pair, composed of a head and a tail vector, employs a classification function, being either softmax or biaffine function, to produce the RE classification result, i.e., the relation type between the two entities. More details about the RE model variations can be found in section 4.2.1.

The RE loss function is also cross-entropy based:

$$loss_{RE} = \sum_{i=1}^{n} -log(\frac{e^{s_{i,q_i}}}{\sum_{j=1}^{k} e^{s_{i,r_j}}}) \quad (2)$$

where n is the total number of relations, q_i is the actual relation label for the i^{th} entity pair, s_{i,q_i} is the linear score for the i^{th} relation belonging to its actual class q_i, k is the number of relation types, and s_{i,r_j} is the linear score for the i^{th} relation belonging to relation class r_j.

The overall joint model loss is derived by summing the NER and RE losses:

$$loss_{joint} = loss_{NER} + loss_{RE} \quad (3)$$

4.2 Model Options

By keeping the NER layers unchanged in this general network architecture, we further experimented with different options for the RE sub-network and evaluated their effects on joint task performance.

4.2.1 RE Sub-network Options

For the RE task, we explored methods of representing entities pairs and classifying their relations, which are rendered as the relation handling and the classification layers in Figure 1. We tested two options, denoted as **re_m1** and **re_m2** respectively. Model option **re_m1** is based on (Bekoulis et al., 2018a,b), which passes entity vectors derived from the NER layer through a fully connected layer for obtaining its head entity representation and similarly through another fully connected layer for tail entity representation, and then adds vectors of a pair of entities to serve as the relation vector for the two paired entities (i.e. head and tail entities):

$$h_{i,j} = h_i + h_j \quad (4)$$

where h_i and h_j are vectors for head and tail entities and $h_{i,j}$ is the resulting vector from summing the two.

We subsequently constructed a fully connected layer to classify the relation vectors. Differing from (Bekoulis et al., 2018a,b), in which the RE classes were assumed to be not mutually exclusive and the RE classification was treated as a multi-label classification task using a sigmoid function, relations in our study are mutually exclusive from each other and henceforth we used a softmax function to classify the relations. Also note that (Bekoulis et al., 2018a,b) used bidirectional LSTM for embedding/encoding and we replaced it with BERT as described in 4.1.

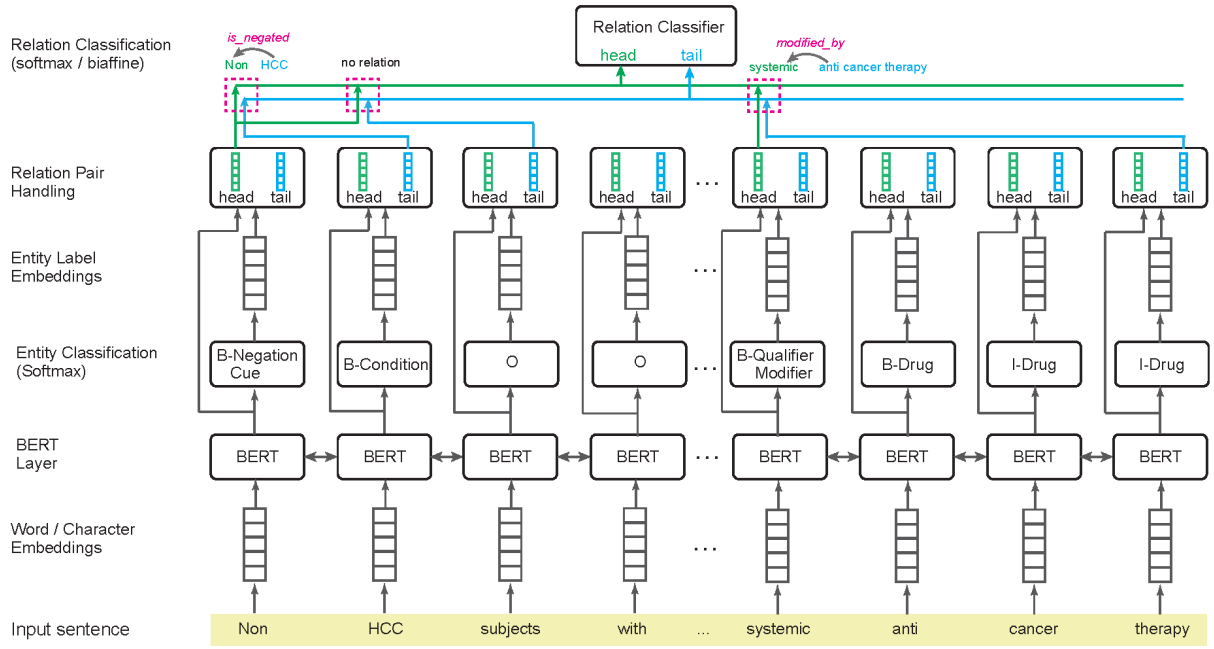

Figure 1: Neural architecture of the joint model for NER and RE tasks.

The other model option, **re_m2**, is similar to the practice in (Giorgi et al., 2019; Nguyen and Verspoor, 2019): we first applied two parallel fully connected layers to derive head and tail entity representation respectively, and then performed biaffine classification using the head and the tail vectors in an entity pair. The biaffine classification function is:

$$biaffine(h_i, h_j) = {h_i}^T U h_j + W(h_i||h_j) + b \quad (5)$$

where h_i and h_j denotes the head and tail entity vectors respectively, U is a tensor of size of $m \times l \times m$, W is a matrix of size of $l * 2m$, with m being the hidden size of the head/tail vector and l being the number of RE labels, $h_i||h_j$ denotes concatenating the two vectors, and b is a bias vector of size of l.

The above biaffine function has a bilinear term ${h_i}^T U h_j$ and a linear term $W(h_i||h_j)$, along with the bias term. We experimented with either including both the bilinear and linear terms (*bilinear+linear*), or only including the bilinear term (*bilinear only*).

4.2.2 RE Negative Sample Construction during Training

A common challenge in RE classification tasks is the overwhelming choices of negative samples. In principle, any entity pair without syntactic relations is a negative sample. To address this challenge, we evaluated two negative sample construction strategies. One strategy is to scan through all the possible entity pairs and mark the pairs without syntactic relations as negative relation samples. Since this option relies on relation information from gold standard data, we denote it as *gs-based*. The other strategy, denoted as *incremental*, incrementally builds negative relation samples based on NER predicted labels, as in (Giorgi et al., 2019; Nguyen and Verspoor, 2019). More specifically, in the *incremental* strategy, an entity pair is included as a negative sample if 1) two entities in this pair are correctly predicted by the NER layer and are without relations, or 2) any of the entities in this pair is incorrectly predicted by the NER. Hence, the former way of constructing negative samples is static as the samples remain unchanged throughout the training, whereas the latter way is dynamic, as whether or not an entity pair is included as a negative sample depends on the NER prediction result during training sessions.

4.2.3 Evaluation Options

We evaluated the joint learning model's performance on NER and RE tasks, by reporting micro-level precision, recall, and f1-measure for both tasks. For RE, we evaluated on relations between gold standard entities without considering NER predicted entities (the *gs-based* option), and also evaluated on relations yielded from NER predicted entities (the *end-to-end* option). In other words, the

gs-based option evaluates RE performance when we know which tokens are actual entities, and the *end-to-end* option evaluates the performance in the scenario when we do not have actual entity information, which is a more realistic scenario when evaluating a RE model's performance in production systems.

4.2.4 Standalone Models

To assess the effects of joint learning options, we built NER and RE standalone models from the corresponding sub-networks in the joint learning architecture for NER and RE tasks separately and evaluate their performances.

For the NER standalone model, following the BERT layer, we added a fully connected layer with softmax classification. For the RE standalone model, instead of having an intermediate NER layer, we appended two parallel fully connected layer directly on the BERT output to derive the head and tail entity representations, and then classified entity pair relations using a softmax function. For standalone model evaluations, we employed the same precision/recall/f1-measures as in the joint models by evaluating against the gold standard (the *gs-based* option).

It is worth noting that in real-world practice, we do not know which tokens are entities so we have to use NER prediction as entity input for RE evaluation. Thus, we included a real-world inspired end-to-end metric for RE that evaluates the performance using NER standalone model prediction as inputs, which effectively takes into account the propagated NER prediction errors (the *end-to-end* evaluation option).

4.2.5 Pre-trained Models

For pre-trained models, we experimented with BERT base, a smaller version of BERT that comprises 110 million parameters, and BioBERT, a model derived from retraining the original BERT using large-scale biomedical texts (Lee et al., 2019). We chose to use the uncased version of BERT base in which all text is lower cased; and since BioBERT is cased only, we used the model with all original cases preserved in text.

4.3 Hyperparameters

We used the same hyperparameter values across all the models as shown in Table 3. For BERT layer hyperparameters we used the same values as in the original BERT model. The models were implemented using the Tensorflow library.

Table 3: Hyperparameter Values.

Hyperparameters	Value
Number of training epochs	20
Learning rate	2×10^{-5}
Training batch size	32
Maximum sequence length	128
NER embedding size	16

5 Results and Analysis

Our results are summarized in Table 4 and we elaborate our finding below.

re_m1 vs. re_m2 RE sub-network option. For the NER task, the four re_m1 models performed similarly to the eight re_m2 models. The highest recall, precision and f1-measure are achieved in re_m2's gs-based negative sampling option (model #12), which performs only marginally better than the other re_m1 and re_m2 models. For the RE task, in the BERT scenario, the re_m2 models greatly outperform the re_m1 models in all three measures (P/R/F). For example, model #5, a re_m2 model using gs-based negative sampling, achieved end-to-end f1-measure of 58.14%, whereas its counterpart, model #1, in the re_m1 model family, has f1-measure of 44.25%, a 13.89% drop from model #5. This result demonstrates that the biaffine classification, the entity pair representation and the classification option used in re_m2, can lead to much better RE performance than softmax classification as used in re_m1. However, for BioBERT pre-trained model, the result is not as decisive: re_m2 does not consistently outperform re_m1. For example, model #15 (re_m2) has better RE performance than model #11 (re_m1) yet model #12 (re_m2) exhibits lower RE performance than model #11 (re_m1).

Biaffine variations for the re_m2 option. Within the re_m2 model, we evaluated the strategies to classify relations using both the bilinear and linear parts of the biaffine function (*bilinear + linear*) or using only the bilinear part (*bilinear only*). The results are exhibited as models #3 to #6 (BERT) and #12 to #15 (BioBERT) in Table 4. The two strategies achieved similar results on the NER task for both BERT and BioBERT cases. For RE end-to-end performance, the *bilinear only* strategy combined with BERT and gs-based negative sampling for training (model #5) achieved the best f1-measure and recall; and the *bilinea + lin-*

ear strategy together with BioBERT and gs-based negative sampling got the highest precision (model #12). Overall, we observed that the biaffine classification options play a less significant role in model performance comparing to other modeling components such like the negative sampling strategies and pre-trained model options.

gs-based vs. incremental RE negative sampling. We tested the two negative sampling strategies on both re_m1 and re_m2 models (models #1 to #6 and #10 to #15 in Table 4). In the case of using pre-trained BERT model, we observed that gs-based negative sampling outperforms the incremental option (models #1 vs. #2, #3 vs. #4, #5 vs. #6) with significant margin. In particular, model #1 exceeded #2 by 4.45% for end-to-end f1-measure, and models #3 and #5 exceeding #4 and #6 by 4.91% and 5.47%, respectively. Interestingly, in contrast, for the BioBERT case, the incremental strategy is superior to the gs-based strategy by an even larger margin, e.g. with model #11 excedding #10 by 19.69%. Therefore the effect of RE negative sampling strategies is jointly determined with the pre-trained model option, and can be significant.

Joint-learning vs. standalone model. Our results show that joint-learning models generally improve RE performance over the standalone RE model but do not significantly affect the NER task (1% drop in f1-measure). because the incremental strategy requires NER net, the standalone RE can only be evaluated using the gs-based strategy, and we had to use gold standard entity information as RE input. The joint-learning models outperform the standalone RE in most of the scenarios when measured with the gs-based evaluation option. When conducting the end-to-end RE task, the joint-learning models exhibit dramatic performance improvement over the standalone models, e.g. f1-measure of 58.14% for model #5 (joint) vs. 48.15% for #9 (standalone) and 55.37% model #15 (joint) vs. 26.41% for #18 (standalone). Despite of the slightly weaker performance in NER task, the great gain in the end-to-end RE task demonstrates that the joint-learning models a better solution in real-world applications.

BERT vs. BioBERT. Comparing between the two pre-trained models, for the NER task, BioBERT yields better result (around 70%) than BERT (around 69%), possibly due to its additional language model pre-trained from biomedical corpus. BERT-based joint-learning models, when using re_m2 negative sampling strategy, outperforms the re_m1 strategy, but this trend does not hold in the BioBERT-based joint-learning models. BioBERT standalone model performance on the RE task is severely impacted by this configuration (model #18). Although BioBERT-based model achieves reasonable performance in some joint-learning (e.g. model #13) strategies, it fails in others (e.g. model #10). These results indicate that joint-learning with BERT is more robust with more stable performance than BioBERT.

In summary, our results demonstrated that joint-learning is a superior strategy, thanks to its steady and significant performance gain in the end-to-end RE task. However, since no model can achieve the best NER and RE performance simultaneously, it is still necessary to balance the two tasks' performances when choosing the proper joint-learning model to prioritize production needs.

6 Conclusion and Future Work

In this reported work, we employed joint-learning models to identify entities and relations in clinical protocols by using pre-trained transformer NLP deep learning models. To the best of our knowledge, this is the first attempt to tackle the NER and RE tasks in a joint and pre-trained deep learning fashion on real-world protocols, which is inherently a corpus of high complexity. Our contribution is three fold: 1) we abstracted a neural network architecture from literature combining pre-trained transformer model with joint learning for NER & RE tasks, 2) we experimented with different model options based on the joint learning network architecture, 3) we examined performance on a complex clinical corpus, which is a less studied but highly impactful domain for such tasks. Our results demonstrated that joint-learning models can greatly improve RE performance over the standalone models despite of a minor decrease in the NER performance. Among all the evaluated joint-learning strategies, the biaffine RE model, gold-standard based negative sampling, together with the BERT pre-trained model, led to generally better performance than other strategies. These results provide evidence on the effectiveness of using joint and deep learning on parsing clinical protocol text, and thus for future work, we will continue exploring more sophisticated joint and multi-task learning network architectures to further enhance the NER and RE parsing performance.

Table 4: NER & RE task performance from joint and standalone models (in percentage).

No.	Pre-trained model	Method	RE sub-network option	RE negative sampling for training	NER Performance			RE Performance					
								gs-based			end-to-end		
					P	R	F	P	R	F	P	R	F
1	bert	Joint model	re_m1	gs-based	66.25	72.71	69.32	70.12	51.93	59.27	47.24	43.68	44.25
2	bert	Joint model	re_m1	incremental	66.73	73.07	69.74	69.29	34.86	43.67	52.44	34.85	39.80
3	bert	Joint model	re_m2, bilinear + linear	gs-based	66.61	73.10	69.69	69.57	64.57	66.86	52.63	64.57	57.89
4	bert	Joint model	re_m2, bilinear + linear	incremental	66.55	72.92	69.58	70.71	51.69	59.38	54.58	51.68	52.98
5	bert	Joint model	re_m2, bilinear only	gs-based	66.46	73.12	69.62	70.47	64.64	67.29	52.97	64.63	**58.14**
6	bert	Joint model	re_m2, bilinear only	incremental	66.72	72.79	69.61	66.72	**72.79**	**69.61**	53.56	51.93	52.67
7	bert	Standalone NER	-	-	67.79	73.19	70.37	-	-	-	-	-	-
8	bert	Standalone RE	linear	gs-based	-	-	-	64.37	48.85	54.89	-	-	-
9	bert	Standalone end to end	linear	-	67.79	73.19	70.37	-	-	-	48.52	48.85	48.15
10	biobert	Joint model	re_m1	gs-based	67.93	73.49	70.58	61.84	45.81	51.53	17.35	45.79	24.59
11	biobert	Joint model	re_m1	incremental	67.70	73.08	70.27	66.74	40.77	49.02	51.73	40.76	44.28
12	biobert	Joint model	re_m2, bilinear + linear	gs-based	68.02	**73.54**	70.66	70.03	66.45	67.93	30.66	**66.45**	41.64
13	biobert	Joint model	re_m2, bilinear + linear	incremental	67.71	73.33	70.38	70.44	55.21	61.50	53.48	55.21	54.22
14	biobert	Joint model	re_m2, bilinear only	gs-based	67.98	73.53	70.63	69.65	66.41	67.82	23.93	66.40	34.73
15	biobert	Joint model	re_m2, bilinear only	incremental	67.70	73.44	70.44	**72.69**	56.19	63.01	**54.66**	56.20	55.37
16	biobert	Standalone NER	-	-	**68.97**	73.50	**71.15**	-	-	-	-	-	-
17	biobert	Standalone RE	linear	gs-based	-	-	-	63.43	52.54	57.05	-	-	-
18	biobert	Standalone end to end	linear	-	**68.97**	73.50	**71.15**	-	-	-	18.03	52.53	26.41

References

Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*.

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction. *Literature review for Language and Statistics II*, 2.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018a. Adversarial training for multi-context joint entity and relation extraction. *arXiv preprint arXiv:1808.06876*.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018b. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45.

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. *arXiv preprint arXiv:1903.10676*.

Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1998. Nymble: a high-performance learning name-finder. *arXiv preprint cmp-lg/9803003*.

Miao Chen, Fang Du, Ganhui Lan, and Victor S Lobanov. 2020. Using pre-trained transformer deep learning models to identify named entities and syntactic relations for clinical protocol analysis. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering (1)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

John Giorgi, Xindi Wang, Nicola Sahar, Won Young Shin, Gary D Bader, and Bo Wang. 2019. End-to-end named entity recognition and relation extraction using pre-trained language models. *arXiv preprint arXiv:1912.13415*.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. 6.2. 2.3 softmax units for multinoulli output distributions. In *Deep Learning.*, pages 180–184. MIT Press.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019a. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Weipeng Huang, Xingyi Cheng, Taifeng Wang, and Wei Chu. 2019b. Bert-based multi-head selection for joint entity-relation extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 713–723. Springer.

Amarin Jettakul, Duangdao Wichadakul, and Peerapon Vateekul. 2019. Relation extraction between bacteria and biotopes from biomedical texts with attention mechanisms and domain-specific contextual representations. *BMC bioinformatics*, 20(1):627.

Dan Jurafsky. 2000. *Speech & language processing*. Pearson Education India.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 917–928.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML proceedings*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Fei Li, Yonghao Jin, Weisong Liu, Bhanu Pratap Singh Rawat, Pengshan Cai, and Hong Yu. 2019. Fine-tuning bidirectional encoder representations from transformers (bert)–based models on large-scale electronic health record notes: An empirical study. *JMIR medical informatics*, 7(3):e14830.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *ICML*, volume 17, pages 591–598.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*.

Dat Quoc Nguyen and Karin Verspoor. 2019. End-to-end neural relation extraction using deep biaffine attention. In *European Conference on Information Retrieval*, pages 729–738. Springer.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019a. Extracting multiple-relations in one-pass with pre-trained transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1371–1377, Florence, Italy. Association for Computational Linguistics.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019b. Extracting multiple-relations in one-pass with pre-trained transformers. *arXiv preprint arXiv:1902.01030*.

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