Biomedical Event Extraction as Sequence Labeling

Alan Ramponi[◇♣] **Rob van der Goot**[♠] **Rosario Lombardo**[♣] **Barbara Plank**[♠]
[◇]Department of Information Engineering and Computer Science, University of Trento, Italy
[♠]Department of Computer Science, IT University of Copenhagen, Denmark
[♣]Fondazione the Microsoft Research – University of Trento
Centre for Computational and Systems Biology (COSBI), Italy
ramponi@cosbi.eu, robv@itu.dk, lombardo@cosbi.eu, bapl@itu.dk

Abstract

We introduce Biomedical Event Extraction as Sequence Labeling (BEESL), a joint end-to-end neural information extraction model. BEESL recasts the task as sequence labeling, taking advantage of a multi-label aware encoding strategy and jointly modeling the intermediate tasks via multi-task learning. BEESL is fast, accurate, end-to-end, and unlike current methods does not require any external knowledge base or preprocessing tools. BEESL outperforms the current best system (Li et al., 2019) on the Genia 2011 benchmark by 1.57% absolute F1 score reaching 60.22% F1, establishing a new state of the art for the task. Importantly, we also provide first results on biomedical event extraction *without* gold entity information. Empirical results show that BEESL's speed and accuracy makes it a viable approach for large-scale real-world scenarios.[1]

1 Introduction

Biomedical event extraction provides invaluable means for assisting domain experts in the curation of knowledge bases and biomolecular pathways (Ananiadou et al., 2010). While the task has received significant attention in research over the last decade, it remains challenging. Progress has been rather stagnating (see Figure 1).

Events are typically highly complex and nested structures, which require deep contextual knowledge to resolve. This is particularly the case for biomedical NLP (Kim et al., 2011), where biomolecular events can be nested (Miwa et al., 2014) and long-distance arguments are frequent (Li et al., 2019). Figure 2 shows an example with four events. Each event consists of an event mention (*trigger*) and one or more *arguments*. For instance, there is a +REGULATION event triggered by the span "induced", with a PROTEIN entity (i.e., "IL-12") as CAUSE and a nested +REGULATION event (i.e., "activation") as THEME. Many state-of-the-art biomedical event extraction systems still work as a pipeline and extract event triggers and their arguments independently (Björne and Salakoski, 2018; Li et al., 2019). They typically employ dependency parsing as features in a CNN model ensemble (Björne and Salakoski, 2018) or in Tree-LSTMs with knowledge bases (Li et al., 2019).

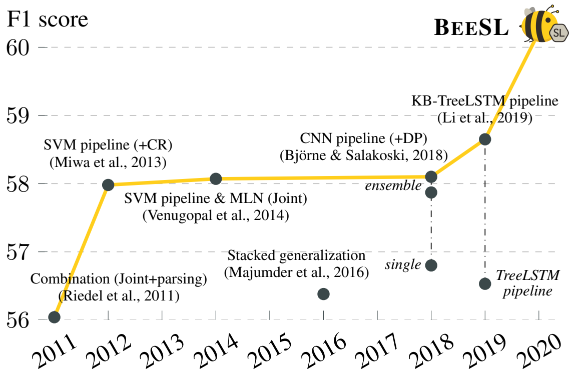

Figure 1: Performance of biomedical event extraction on the BioNLP Genia 2011 test set over time.

We propose a new approach for biomedical event extraction by casting it as a sequence labeling task (BEESL). Our approach is conceptually simple: we convert the event structures into a representation suitable for sequence labeling, and leverage a multi-label aware decoder with BERT (Devlin et al., 2019) in a multi-task sequence labeling model. This reduces the problem to predicting a structured output for an input sequence to word-level tagging decisions. Compared to previous alternatives (cf. Section 7) which cast event extraction as syntactic or semantic tree- or graph-parsing task, this leads to a faster, joint model which also mitigates error propagation of locally-optimized classifier pipelines (Björne and Salakoski, 2018;

[1] The source code is available at https://github.com/cosbi-research/beesl.

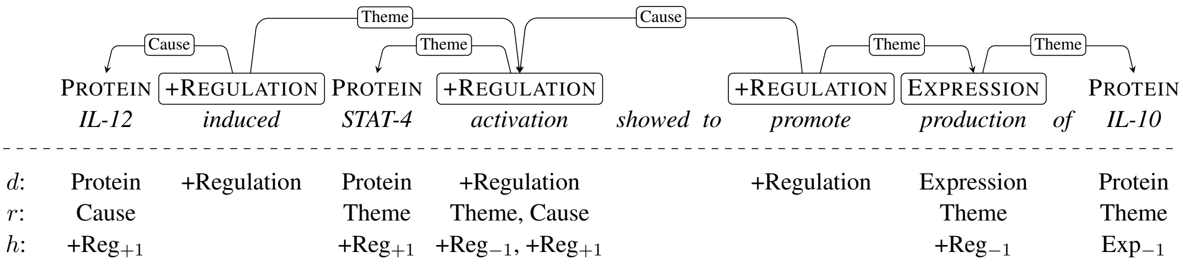

Figure 2: Top: a text excerpt with four biomedical events. Above the text (*italicized*), mentions (triggers inside rounded boxes, and entities without rounded boxes) and argument roles are indicated. Bottom: our proposed encoding, where d, r and h represent the label parts for dependents, relations, and heads, respectively.

Li et al., 2019). Our empirical evaluation shows the effectiveness of BEESL for biomedical event extraction. A quantitative and qualitative analysis shows that BEESL is fast and effective. Despite the model's simplicity, BEESL outperforms the previous best model (Li et al., 2019) on most event categories.

Contributions To the best of our knowledge, we are the first to cast biomedical event extraction as sequence labeling. We demonstrate that BEESL is an attractive and efficient solution to extract biomedical events. We evaluate it on the BioNLP Genia 2011 benchmark, obtaining a new state of the art (cf. Figure 1), while gaining on efficiency. We additionally provide empirical results of the impact of alternative multi-task encodings, and to the best of our knowledge, the first results of biomedical event extraction *without* assuming gold entities.

2 Encoding Event Structures

This section introduces the event structures and how we encode them for sequence labeling.

2.1 Event structures

Events are structured representations which comprise multiple information units (Figure 2, top). An event is anchored to a *trigger*, a text span which indicates the presence of an event (Figure 2, rounded boxes). Each event has one or more arguments, namely *entities* or other events (Figure 2, end of arrows), which are assigned a *role* in the event (Figure 2, labels on arrows). For example, an EXPRESSION event is indicated in Figure 2 at "production" involving the PROTEIN "IL-10" as its argument. Nested structures are possible and frequent. For instance, the +REGULATION event centered on "activation" is both argument of the "induced"-anchored event as well as the "promote"-anchored event.

2.2 Sequence labeling encoding

Given $[x_1, ..., x_n]$ a sequence of n tokens, we encode event structures as token-level labels $[y_1, ..., y_n]$, to reduce the task to a sequence labeling problem. Adopting dependency parsing terminology, we encode the label y_i for each token x_i as a tuple $\langle d, r, h \rangle$, where d is the *dependent* and refers to the token and its mention type (either trigger, entity, or nothing), r is the *relation* and used to refer to its role, and *head* (h) denotes the event the token refers to (Figure 2, bottom). In more detail, to discriminate event heads with the same type in text, we encode the heads h as *relative head mention position*.[2] For instance, h = +REG$_{+1}$ means the head is the first +REGULATION on the right of d in the relative surface order, whereas h = +REG$_{-2}$ means it is the second +REGULATION on the left. In Figure 2 the label for "production" is $\langle$EXPRESSION, THEME, +REG$_{-1}\rangle$, denoting the token is an EXPRESSION trigger, THEME of the first +REGULATION event on the left.

As opposed to dependency parsing, tokens may have zero or multiple roots, and thus multiple heads and relations. This poses additional challenges. For instance, the "activation"-anchored event (Figure 2) is both THEME and CAUSE of "induced"- and "promote"-anchored event heads, respectively. As a result, both r and h are multi-label, and the label for "activation" is encoded as $\langle$+REGULATION, [THEME, CAUSE], [+REG$_{-1}$, +REG$_{+1}$]$\rangle$, where the order of r and h items is preserved.

3 Event Extraction as Sequence Labeling

Formally, we aim to learn a function $f : X \mapsto Y$ that assigns each token x_i a structured label y_i, i.e.,

[2] In preliminary experiments we found this mitigates the label sparsity problem of other positional encodings, e.g., relative positional encoding (Strzyz et al., 2019). We additionally found relative head mention positions ≥ 2 are rare in our data.

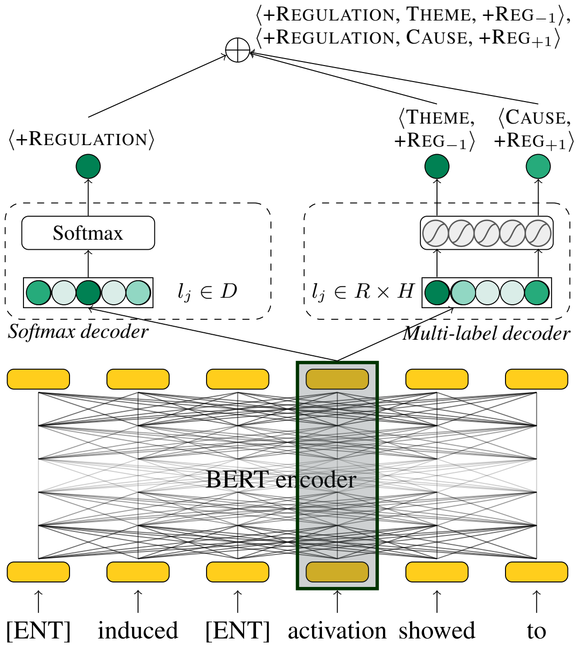

Figure 3: BEESL uses a multi-task multi-label model, using a BERT encoder with layer attention, and dedicated decoders for predicting the labels for each label sub-space, which are trivially merged.

$\langle d, r, h\rangle$. A straightforward solution is to predict the label y_i as an atomic entity (i.e., single label) in a single-task model. For BEESL, we instead propose to use multi-task learning (MTL) which allows to learn interdependencies while cutting down the label space, paired with multi-label prediction.

An overview of BEESL is shown in Figure 3. We use BERT (Devlin et al., 2019) as encoder, pretrained on biomedical texts (Section 4). We mask entity spans for better generalization (Alt et al., 2019). The first WordPiece (Schuster and Nakajima, 2012) of each token x_i is used for prediction, where the contextual hidden representation e_i of the token x_i is encoded with layer-wise attention over the BERT layers, similarly to (Peters et al., 2018; Kondratyuk and Straka, 2019). As decoders, we use standard softmax with a cross entropy loss unless otherwise specified, and introduce a multi-label decoder (Section 3.2) (Figure 3, upper right).

We empirically evaluate both single-task and multi-task setups, including several MTL encoding alternatives, discussing their limitations and benefits. In the following, we first introduce the multi-task setups, and then multi-label decoding.

3.1 Multi-task strategies

We denote the label spaces for each component of the labels as $d_i \in D$, $r_i \in R$, and $h_i \in H$. Further, we use $\mathcal{L}$ to refer to the maximum label space size.

Single-task A single-task (ST) setup is used as a baseline. It predicts a single label $y_i = \langle d, r, h\rangle$ for each input token x_i. The label space is up to $\mathcal{L} = |D| \times |R| \times |H|$.

Multi-task The label y_i for each token x_i is decomposed into parts (hereafter, sub-labels), each treated as a prediction task. The decomposition of the label space allows each sub-label space to be framed as a different task with its own private decoder, mitigating the output space sparsity (Vilares et al., 2019). Depending on the decomposition of the label $y_i = \langle d, r, h\rangle$, we have four multi-task learning options (pairs of tasks, or each subpart as a task, respectively) with the following properties:

1. $\langle d\rangle, \langle r, h\rangle$: up to $\mathcal{L} = |D| + |R| \times |H|$;
2. $\langle d, r\rangle, \langle h\rangle$: up to $\mathcal{L} = |D| \times |R| + |H|$;
3. $\langle d, h\rangle, \langle r\rangle$: up to $\mathcal{L} = |D| \times |H| + |R|$;
4. $\langle d\rangle, \langle r\rangle, \langle h\rangle$: up to $\mathcal{L} = |D| + |R| + |H|$.

Option 4 encodes each subpart as its own task. While this leads to the smallest label space, it decouples the problem into 3 separate tasks. Options 1-3 are pair-wise task setups. We hypothesize that BEESL benefits from disentangling mention detection from head labeling (option 1).

As illustrated in Figure 3, BEESL uses the predicted sub-labels to form the complete label tuple $\hat{y}_i = \langle \hat{d}, \hat{r}, \hat{h}\rangle$. In case r and h belong to different sub-label spaces (as is possible in options 2-4), we require that both predictions $\hat{r}$ and $\hat{h}$ are present (non-empty) to ensure well-formedness. This is a downside of these alternative options 2-4, as we will see empirically (Section 5).

During training, the MTL loss is computed as $L = \sum_t \lambda_t L_t$, where L_t is the loss for each task t, given by the respective decoder (see also Section 3.2), with λ_t a task-specific weighting parameter. In our experiments we kept $\lambda = 1.0$ for all, since preliminary experiments showed weighting sub-tasks differently was not beneficial. In the single-task setup, the loss reduces to $L = L_t$.

3.2 Multi-label decoder

The multi-label decoder is designed to handle multiple labels per token, thus being suitable for predicting relations and heads. Given a task with $l_j \in L$ labels, it models $P(l_j|e_i)$ for each label l_j. Differently from the single-label decoder, each label

Item	Train	Dev	Test
Documents	908	259	347
Sentences	8,664	2,888	3,363
Tokens	230,737	74,334	90,091
Entities	11,625	4,690	5,301
Events	10,310	3,250	4,487

Table 1: Statistics of the Genia 2011 event dataset.

is predicted with a sigmoid, where all contribute equally to the loss. Given the probabilities $P(l_j|e_i)$ for the $l_j \in L$ labels and a threshold τ, the token x_i is assigned all the labels l_j with probability $P(l_j|e_i) \geq \tau$. If no $P(l_j|e_i) \geq \tau$ is found, we take the highest scoring label l_j (which may also be empty) as a fallback.[3] We employ a binary cross-entropy loss, averaged across all batches.

4 Experimental Setup

We evaluate BEESL on the Genia 2011 benchmark (Kim et al., 2011), which comprises both abstracts and full-texts. The corpus consists of annotations for PROTEIN entities and 9 fine-grained event types. The Genia event extraction tasks expect both texts and entities as input, and complete events need to be predicted. Statistics on the dataset are shown in Table 1.

Event types can be categorized into simple, binding and complex events, related to the number and types of arguments. *Simple events* require a THEME only, *binding events* require one or more THEME arguments, while *complex events* take both THEME and CAUSE arguments, where both can in turn be other events, resulting in nested structures. Björne and Salakoski (2011) estimated that 37.2% of the events in the data are nested. We refer the reader to Appendix A.1 for formal event definitions.

BEESL is based on MaChAmp (van der Goot et al., 2020), a toolkit for multi-task learning and fine-tuning of BERT-like models. We extend MaChAmp to also handle multi-label sequence labeling. We experiment with BEESL in single- and different multi-task setups.

After sequence labeling, token-level labels are converted into the official BioNLP-ST standoff format for evaluation (Kim et al., 2011). We simply split the event arguments based on their formal definition, producing complete structures (e.g., an EXPRESSION event with k THEME arguments is split into k EXPRESSION events, with one THEME each). Similarly to previous work, we focus on sentence-level events. We used BioBERT-Base 1.1 as our BERT model for experiments, since it provides state-of-the-art performance across multiple biomedical information extraction tasks (Lee et al., 2019). For multi-label decoding, we tune the threshold τ for each setup (yielding $\tau_{MT} = 0.5$ and $\tau_{ST} = 0.7$). Other hyper-parameter values and tuning details are provided in Appendix A.2.

Evaluation In line with previous work, we evaluate BEESL in terms of precision (P), recall (R), and F1 score according to the approximate recursive span matching criterion (Kim et al., 2011) using the official BioNLP online evaluation service.[4] For early stopping during training, we employ the simpler span-based F1 score (as used in named entity recognition) as our proxy metric. We found it highly correlates with the approximate recursive span based F1 official metric.

No gold entities In biomedical event extraction, entities are typically given in advance. To evaluate BEESL in a setup with *predicted* entities (Section 6.3), we firstly employ our model as single-task sequence labeler for BIO-tagged entity mentions using default settings and a standard CRF decoder (Gardner et al., 2018). Note that for comparison purposes in all other experiments we assume entity mentions are gold-tagged. Then, we evaluate BEESL with raw texts and *predicted* entities as input, thus indirectly penalizing events that take over-predicted entities or that miss entities since they are under-predicted.

5 Results

First, we evaluate the MTL and multi-label decoding strategies on the development set to determine the best setup (Sections 5.1, 5.2). Then, we compare BEESL to the results obtained by the top performing systems on the official test set (Section 5.3). Finally, we gauge its speed (Section 5.4).

5.1 Multi-task settings

Table 3 (top) summarizes the main results for the MTL experiments. They confirm our hypothesis that $\langle d \rangle, \langle r, h \rangle$ (option 1) is the most viable representation; it leads to the highest F1 score, largely

[3] In case $\tau = 0 \vee \tau = 1$, we adopt the same strategy, since all or no labels would be potentially predicted, respectively.

[4] http://bionlp-st.dbcls.jp/GE/2011/eval-test/.

Work	Method	P	R	F1
Riedel et al. (2011)	FAUST – Model combination (joint+parsing)	64.75	49.41	56.04
Miwa et al. (2012)	EventMine – SVM pipeline (+coref)	63.48	53.35	57.98
Venugopal et al. (2014)	BioMLN – SVM pipeline & MLN (joint)	63.61	53.42	58.07
Majumder et al. (2016)	Stacked generalization	66.46	48.96	56.38
Björne and Salakoski (2018)	TEES – CNN pipeline (single model)	64.86	50.53	56.80
Björne and Salakoski (2018)	TEES – CNN pipeline (5x ensemble)	68.76	49.97	57.87
Björne and Salakoski (2018)*	TEES – CNN pipeline (mixed 5x ensemble)	69.45	49.94	58.10
Li et al. (2019)	BiLSTM pipeline	62.18	48.44	54.46
Li et al. (2019)	Tree-LSTM pipeline	64.56	50.28	56.53
Li et al. (2019)	KB-driven Tree-LSTM pipeline	67.01	52.14	58.65
BEESL	Multi-task neural sequence labeling	69.72	53.00	**60.22**

Table 2: Performance comparison on the test set of BioNLP Genia 2011. *indicates that the system was trained on training plus part of development data. BEESL uses the official training portion only. Top: traditional ML systems; Middle: state-of-the-art neural systems; Bottom: proposed multi-task sequence labeling system.

Multi-task	P	R	F1
$\langle d \rangle, \langle r, h \rangle$	71.28	55.44	**62.37**
$\langle d, r \rangle, \langle h \rangle$	72.35	51.31	60.04
$\langle d, h \rangle, \langle r \rangle$	73.51	49.49	59.16
$\langle d \rangle, \langle r \rangle, \langle h \rangle$	73.05	51.34	60.30
Multi-label	**P**	**R**	**F1**
BEESL$_{ST}$	73.30	52.42	61.13
with multi-label	71.74	56.71	63.34
BEESL$_{MT}$	71.28	55.44	62.37
with multi-label	71.84	59.42	**65.04**

Table 3: Performance of diverse settings for BEESL (multi-task and multi-label) on the development set.

outperforming the other MTL options, particularly in recall. These results show that a multi-task setup with separate tasks for mention detection and head labeling, respectively, is the most useful. Option 1, i.e., $\langle d \rangle, \langle r, h \rangle$ defaults to the multi-task option for BEESL (Figure 3) used in the following experiments.

5.2 Adding the multi-label decoder

We evaluate the multi-label decoder for both single-task (BEESL$_{ST}$) and multi-task (BEESL$_{MT}$) setups (Table 3, bottom). Multi-label decoding is beneficial, as the data contains many multi-headed tokens, and modeling them improves both setups. Single task performance increases substantially, from 61.13 to 63.34 F1 score. Similar significant performance gains are observed for multi-task learning, from 62.37 to 65.04 F1 score. Regardless of the multi-label modeling, the multi-task setup provides the highest overall performance.

5.3 Comparison to the state of the art

We now compare the multi-task multi-label BEESL to the top performing systems (hereafter, simply BEESL). As shown in Table 2, BEESL outperforms the state-of-the-art by a large margin, i.e., an absolute improvement of 1.57 points in F1 score over the KB-Tree LSTM model (Li et al., 2019) (hereafter, KBTL). It improves over both precision and recall, and yields a new state of the art with an F1 score of 60.22%, yet being conceptually simple.

Table 4 compares F1 scores of BEESL to the previous best model on a per-event level (precision and recall are provided in Appendix A.3). BEESL outperforms the KBTL approach (Li et al., 2019) overall on 7 out of the 9 event types. From a coarse-grained perspective, BEESL outperforms KBTL on simple, binding, and complex event categories. Particularly, improvements over KBTL on simple events are as large as +13% F1 score. Furthermore, noticeable are also the improvements for binding and nested, complex events, for which our model achieves 50.19% and 48.32% F1 score. From a closer look, the recall of BEESL on simple events is substantially higher than KBTL, which ease a correct identification of complex events.

Next, we look at performance per text type (i.e., abstract and full-text subsets). BEESL achieves 62.14% F1 score on abstracts-only documents,

Event type	BeeSL	KBTL
Simple events	**79.31**	78.73
Gene expression	**80.90**	80.28
Transcription	69.46	**75.39**
Protein catabolism	**74.07**	60.87
Phosphorylation	**89.52**	84.36
Localization	**69.51**	68.47
Binding	**50.19**	44.10
Complex events	**48.32**	47.72
Regulation	**45.90**	43.52
Positive regulation	**49.41**	48.26
Negative regulation	47.17	**49.02**
All events	**60.22**	58.65

Table 4: Per-event performance of BeeSL and KBTL (KB-driven TreeLSTM) (Li et al., 2019) on the test set.

and 55.59% F1 score on full-texts. This confirms that full-texts are harder to process than abstracts, due to the differences in structural and content aspects (Cohen et al., 2010).

To sum up, BeeSL handles events well, and unlike most prior work, does not use knowledge bases or dependency parsers as pre-processing step. BeeSL uses multi-task learning with a contextual encoder and multi-label aware decoding, herewith bringing progress to the biomedical event extraction task as illustrated in Figure 1.

5.4 Speed comparison

We compare BeeSL to TEES, the Turku Event Extraction System (Björne and Salakoski, 2018) to compare their speed at inference time on commodity hardware. TEES is the 2nd top-performing system (Figure 1), and its code is freely available. To the best of our knowledge, the source code of (Li et al., 2019) is not yet available.

Results in Table 5 show that BeeSL is ∼2x faster and ∼5x faster on a consumer grade CPU[5] than TEES single and ensemble system, respectively. In terms of sentences per minute, BeeSL processes ∼500 sents/min compared to 255 sents/min and 101 sents/min in TEES single (3.42% lower F1) and ensemble (2.12% lower F1), respectively.

	sents/min
TEES (*single*)	$255_{\pm 1}$
TEES (*ensemble*)	$101_{\pm 1}$
BeeSL	$499_{\pm 3}$

Table 5: Speed comparison to TEES (Björne and Salakoski, 2018) single and ensemble models at inference time. Results are sents/min, averaged over 5 runs.

Setting	P	R	F1
BeeSL	71.84	59.42	**65.04**
– multi-task	71.66	56.95	63.47
– multi-label	74.28	52.39	61.44

Table 6: Ablation study on BeeSL when removing the multi-task capability (i.e., replacing MTL with independent classifiers) and the multi-label handling.

6 Analysis and Discussion

To gain insights about BeeSL, we shed more light on several aspects. Firstly, we analyze how much BeeSL gains from multi-task learning, compared to using a powerful contextualized BERT encoder alone in a single-task learning setup and a formulation with two independent classifiers (Section 6.1). Then, we quantify the stability of the threshold τ of the multi-label decoder (Section 6.2). We also aim to get deeper insight on model performance without gold entities (Section 6.3), and qualitatively study the sources of prediction errors of BeeSL (Section 6.4).

6.1 How important is multi-task learning?

As opposed to running one single model which models $\langle d \rangle$ and $\langle r, h \rangle$ jointly in a multi-task setup, we also compare to single-task (ST) and an experiment in which we formulate two classifiers which predict the two labels from the best MTL setup separately. This allows us to gauge the effectiveness of the multi-task learning approach compared to local classifiers which use strong BERT-based encoding, and compared to predicting an atomic label in ST.

Results in Table 6 confirm that leveraging a shared encoder and multi-task learning for both triggers and heads is crucial. Without multi-task learning and multi-label decoding, the F1 score drops to 61.44 (independent classifiers) and 61.13

[5] Intel Core i5-6360U (2 cores).

Setting	F1	Δ
BEESL$_{ST}$ (multi-label)	63.34	
with best-only prediction	62.87	-0.47
BEESL$_{MT}$ (multi-label)	**65.04**	
with best-only prediction	64.54	-0.50

Table 7: Ablation study on the threshold τ of the multi-label decoder ("with best-only predicion": $\tau = 1.0$).

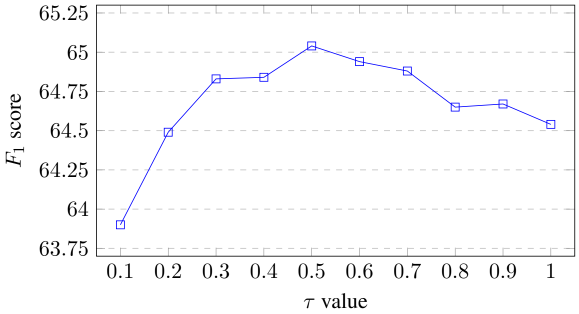

Figure 4: Stability of the threshold τ. Values in the range 0.3-0.7 only minimally alter BEESL scores.

(ST setup, BEESL$_{ST}$ in Table 3). Adding multi-label decoding helps, as expected. However, the full power of BEESL is only achieved by using *both* the multi-task and the multi-label approach, which leads to the novel state of the art.

6.2 How brittle is BEESL to the threshold τ?

As shown in Table 3, using a multi-label decoder largely increases the performance of a system with a single-label decoder (from 62.37 to 65.04 F1 score). However, what is left is how much the threshold τ impacts the performance. To get insights on it, we firstly performed an ablation study setting $\tau = 1.0$. As introduced in Section 3.2, this reduces to predicting the highest scoring label only – however, in a reduced label space induced by the multi-label aware decoder. We found only part of the improvement is due to the threshold τ in both multi-task and single-task settings (+0.50% and +0.47%, respectively) (Table 7).

Moreover, we evaluated BEESL with different τ values. As shown in Figure 4, a threshold in the range 0.3-0.7 only marginally alters the results, which are still better than predicting the highest scoring label only ($\tau = 1.0$).

6.3 What is the effect of using gold entities?

	P	R	F1
BEESL	71.84	59.42	**65.04**
– gold entities	66.15	54.09	59.51

Table 8: Performance of BEESL with *no* gold entities.

Error type	Fraction
Trigger	
Under-prediction	31.43%
Over-prediction	28.57%
Wrong type	10.00%
Argument	
Under-prediction	22.86%
Over-prediction	7.14%
Wrong type	0.00%

Table 9: Error analysis on a random sample of 30 documents from the development set.

The standard in biomedical event extraction is to evaluate the performance of a system on gold entities. In real-world situations it is unlikely that the data is annotated for entities. We believe it is important to estimate the impact non-gold entities have on system performance (hereafter, *silver entities*). The performance of the entity prediction on the development set is 87.95 span-based F1 score.

The results on the event extraction task using silver entities are shown in Table 8. The overall drop in F1 amounts to around 5%, and it is well-balanced across precision and recall. This shows that BEESL's performance is clearly affected, but that the system is relatively robust to noisy, non-gold *silver entities*. We believe that this performance gap can be further minimized by using jackknifing (Agić and Schluter, 2017) to reduce data mismatch, however, this requires to align the predicted entities with the existing events in the training data, which is non-trivial, and we leave this for future work.

6.4 What are the sources of errors?

We randomly sampled 30 documents (comprising 168 gold events) from the development set for a manual scrutiny for sources of errors. We classified errors into two broad categories, namely trigger and argument errors. Further, we classify them in fine-grained categories based on the type of error, namely under-prediction, over-prediction, and wrong type. Table 9 summarizes the results.

We notice the largest fraction of errors is due to trigger errors. From a closer look, under-predicted triggers account for 31.43% of the total, whereas over-predicted triggers for 28.57%. We investigated the reason for these errors, finding that over-predicted triggers are often due to generic words used very frequently to indicate specific trigger types. For instance, BEESL identifies the +REGULATION event anchored at "activated" in the following sentence: "*Tax [...] maximally activated HTLV-I-LTR-CAT and kappa B-fos-CA*" albeit the gold standard does not contain the event in this instance. However, from a semantic point of view we believe these errors are acceptable. Other cases include the words such as "detected" and "influences", which are often used as EXPRESSION and REGULATION event triggers, respectively.

Under-prediction of triggers is instead due to a variety of reasons. Both rare words (e.g., a +REGULATION event centered on "co-transfected") and uncertain events account for a large fraction of this error type. An example of uncertain event is represented by the +REGULATION trigger "importance" in the sentence "*[...] importance of NF-kappa B in LT gene expression*", that BEESL does not predict.

Wrongly typed triggers represent only 10% of the errors. An example is represented by ambiguous trigger types. In the sentence "*T cells upregulates A3G mRNA levels*", BEESL classifies "levels" as an EXPRESSION trigger, while the gold annotation indicates it is a TRANSCRIPTION trigger. By a closer look, we found some triggers in the corpora are annotated as EXPRESSION and TRANSCRIPTION types interchangeably. This is due to the fact a TRANSCRIPTION is a gene EXPRESSION.

Regarding the identification of arguments, over-predictions are quite uncommon. If erroneous, the main error we found may benefit from syntactic information, which we aim to integrate in a multi-task setup in future work. We found no misclassification of arguments in our document samples. Under-prediction of arguments are instead mostly due to under-predicted events.

7 Related Work

Biomedical event extraction has a long-standing tradition (Riedel et al., 2011; Miwa et al., 2012; Vlachos and Craven, 2012; Venugopal et al., 2014; Majumder et al., 2016). Current work has explored neural methods and uses multiple classification stages. Namely, first identifying trigger mentions, and then evaluating all entity pairs (Li et al., 2019; Björne and Salakoski, 2018). They come with the shortcomings of traditional pipeline methods. Many studies use dependency parsers to obtain features or for guidance of Tree-LSTMs (Li et al., 2019; Björne and Salakoski, 2018).

Recent work in syntactic parsing has shown that reducing parsing to sequence labeling is a viable alternative for both constituent and dependency parsing (Spoustová and Spousta, 2010; Gómez-Rodríguez and Vilares, 2018; Strzyz et al., 2019), which we took as inspiration. Moreover, earlier work framed biomedical event extraction as syntactic and semantic tree- or graph-parsing (McClosky et al., 2011; Rao et al., 2017). In particular, McClosky et al. (2011) do dependency parsing, followed by a second-stage parse reranker model for event extraction, and Rao et al. (2017) cast the problem as subgraph identification problem.

Joint learning for biomedical event extraction was explored in early work (Riedel and McCallum, 2011; Venugopal et al., 2014; Vlachos and Craven, 2012). Contemporary to our work, a very recent study proposes oneIE, a joint learning model for event extraction (Lin et al., 2020). It proposes a single end-to-end model for event extraction using 4 stages, paired with a beam search, obtaining good results on ACE data. Processing multiple heads has previously been done for relation extraction using multi-head selection (Bekoulis et al., 2018a,b), and sequence labeling has been employed for joint entity and relation classification (Dai et al., 2019) with inter-token attention. We employ it at the token-level for multi-label sequence labeling.

8 Conclusion

This paper proposes BEESL, a new end-to-end biomedical event extraction system which is both efficient and accurate. BEESL is broadly applicable to event extraction and other tasks that can be recast as sequence labeling. The system's strength comes from the joint multi-task modeling paired with multi-label decoding, which aids interdependencies between the tasks and is superior to alternative decoders based on strong contextualized BERT embeddings. BEESL is fast, and achieves state-of-the-art performance on the Genia 2011 event extraction benchmark without the need of external tools for features and resources such as knowledge bases. Our analysis shows that BEESL works very well across event types.

We release the code freely, to foster research on using BEESL for other NLP tasks as well, e.g., enhanced dependency parsing, fine-grained named entity recognition, and semantic parsing.

Acknowledgments

This research was supported by Fondazione the Microsoft Research – University of Trento Centre for Computational and Systems Biology, Italy, an Amazon Research Award, Independent Research Fund Denmark (Sapere Aude grant 9063-00077B), and NVIDIA corporation for sponsoring Titan GPUs.

References

Željko Agić and Natalie Schluter. 2017. How (not) to train a dependency parser: The curious case of jackknifing part-of-speech taggers. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 679–684, Vancouver, Canada. Association for Computational Linguistics.

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Improving relation extraction by pre-trained language representations. In *Proceedings of AKBC 2019*.

Sophia Ananiadou, Sampo Pyysalo, Jun'ichi Tsujii, and Douglas Kell. 2010. Event extraction for systems biology by text mining the literature. *Trends in biotechnology*, 28(7):381–390.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018a. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2830–2836, Brussels, Belgium. Association for Computational Linguistics.

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.

Jari Björne and Tapio Salakoski. 2011. Generalizing biomedical event extraction. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 183–191, Portland, Oregon, USA. Association for Computational Linguistics.

Jari Björne and Tapio Salakoski. 2018. Biomedical event extraction using convolutional neural networks and dependency parsing. In *Proceedings of the BioNLP 2018 workshop*, pages 98–108, Melbourne, Australia. Association for Computational Linguistics.

K Bretonnel Cohen, Helen L Johnson, Karin Verspoor, Christophe Roeder, and Lawrence E Hunter. 2010. The structural and content aspects of abstracts versus bodies of full text journal articles are different. *BMC bioinformatics*, 11(1):492.

Dai Dai, Xinyan Xiao, Yajuan Lyu, Shan Dou, Qiaoqiao She, and Haifeng Wang. 2019. Joint extraction of entities and overlapping relations using position-attentive sequence labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6300–6308.

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

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Carlos Gómez-Rodríguez and David Vilares. 2018. Constituent parsing as sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1314–1324, Brussels, Belgium. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, and Barbara Plank. 2020. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. *arXiv preprint arXiv:2005.14672*.

Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. 2011. Overview of Genia event task in BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 7–15, Portland, Oregon, USA. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

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*, 36(4):1234–1240.

Diya Li, Lifu Huang, Heng Ji, and Jiawei Han. 2019. Biomedical event extraction based on knowledge-driven tree-LSTM. 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 1421–1430, Minneapolis, Minnesota. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Amit Majumder, Asif Ekbal, and Sudip Kumar Naskar. 2016. Biomolecular event extraction using a stacked generalization based classifier. In *Proceedings of the 13th International Conference on Natural Language Processing*, pages 55–64, Varanasi, India. NLP Association of India.

David McClosky, Mihai Surdeanu, and Christopher Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626–1635, Portland, Oregon, USA. Association for Computational Linguistics.

Makoto Miwa, Paul Thompson, and Sophia Ananiadou. 2012. Boosting automatic event extraction from the literature using domain adaptation and coreference resolution. *Bioinformatics*, 28(13):1759–1765.

Makoto Miwa, Paul Thompson, Ioannis Korkontzelos, and Sophia Ananiadou. 2014. Comparable study of event extraction in newswire and biomedical domains. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2270–2279, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sudha Rao, Daniel Marcu, Kevin Knight, and Hal Daumé III. 2017. Biomedical event extraction using abstract meaning representation. In *BioNLP 2017*, pages 126–135, Vancouver, Canada,. Association for Computational Linguistics.

Sebastian Riedel and Andrew McCallum. 2011. Robust biomedical event extraction with dual decomposition and minimal domain adaptation. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 46–50, Portland, Oregon, USA. Association for Computational Linguistics.

Sebastian Riedel, David McClosky, Mihai Surdeanu, Andrew McCallum, and Christopher D. Manning. 2011. Model combination for event extraction in BioNLP 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 51–55, Portland, Oregon, USA. Association for Computational Linguistics.

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

Drahomíra Spoustová and Miroslav Spousta. 2010. Dependency parsing as a sequence labeling task. *The Prague Bulletin of Mathematical Linguistics*, 94(1):7–14.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2019. Viable dependency parsing as sequence labeling. 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 717–723, Minneapolis, Minnesota. Association for Computational Linguistics.

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 2014. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 831–843, Doha, Qatar. Association for Computational Linguistics.

David Vilares, Mostafa Abdou, and Anders Søgaard. 2019. Better, faster, stronger sequence tagging constituent parsers. 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 3372–3383, Minneapolis, Minnesota. Association for Computational Linguistics.

Andreas Vlachos and Mark Craven. 2012. Biomedical event extraction from abstracts and full papers using search-based structured prediction. *BMC Bioinformatics*, 13(11):1759–1765.

A Appendix

A.1 Data and formal event definitions

Events on the Genia 2011 benchmark follow the formal specification detailed in Table 10. The full data can be downloaded from the official portal.[6]

Event type	Arguments
Simple events	
Gene expression	Theme(P)
Transcription	Theme(P)
Protein catabolism	Theme(P)
Phosphorylation	Theme(P)
Localization	Theme(P)
Binding	Theme(P)+
Complex events	
Regulation	Theme(P/E), Cause(P/E)
Positive regulation	Theme(P/E), Cause(P/E)
Negative regulation	Theme(P/E), Cause(P/E)

Table 10: Formal definition of events. P: PROTEIN, E: any event type, +: 1 or more arguments.

A.2 Hyper-parameters

The list of hyper-parameter values and the search space are presented in Table 11, whereas the number of trainable parameters in BEESL is $\approx 110M$. For tuning, we started from the values reported in previous works on multi-task learning for NLP evaluation benchmarks, e.g., UDify (Kondratyuk and Straka, 2019). We performed 32 search trials via grid search, in which "batch size" and "base learning rate" have been coupled – $(32, 1e^{-}3)$ and $(64, 1e^{-}2)$. Additional 9 search trials have been performed for threshold τ selection for the BEESL multi-task multi-label model. We used the official approximate recursive span matching based F1 score for model selection, whereas the sum of span-based F1 scores of the tasks was employed to determine early stopping of the training process.

A.3 Miscellaneous

Technical details Texts have been tokenized and segmented using scispaCy 0.2.4 (Neumann et al., 2019). In our data it is uncommon that multiple contiguous triggers have the same type, so BIO encoding is not needed. In the rare case of overlapping event triggers of different types, we create a single label d concatenating their types. Similarly to previous work, for BINDING events with multiple THEME arguments we employ a simple heuristic to convert them into the BioNLP-ST standoff format (Vlachos and Craven, 2012). For speed experiments with TEES (Björne and Salakoski, 2018), we removed extra modules for a fair comparison.

Hyper-parameter	Value	Space
Optimizer	Adam	
β_1, β_2	0.9,0.99	
Weight decay	0.01	
Gradient clipping	10	
Dropout	0.5	0.1, 0.3, 0.5
BERT dropout	0.1	0.1, 0.2
Mask probability	0.1	0.1, 0.15, 0.2
Layer dropout	0.1	
Batch size	64	32, 64
Base learning rate	$1e^{-}2$	$1e^{-}3$, $1e^{-}2$
BERT learning rate	$5e^{-}5$	
Epochs	50	
Patience	5	
Multi-label threshold	0.5	0.1, 0.2, ..., 1.0

Table 11: Hyper-parameter values and search space.

Detailed per-event scores We present in Table 12 a complementary view of scores (i.e., with precision and recall) of BEESL and the previous state of the art (Li et al., 2019) on a per-event level.

	BEESL			KBTL		
Event type	**P**	**R**	**F1**	**P**	**R**	**F1**
Simple events	84.17	**74.98**	**79.31**	**85.95**	72.62	78.73
Gene expression	84.55	**77.54**	**80.90**	**87.24**	74.35	80.28
Transcription	72.50	66.67	69.46	**82.31**	**69.54**	**75.39**
Protein catabolism	83.33	**66.67**	**74.07**	**87.50**	46.67	60.87
Phosphorylation	**94.05**	**85.41**	**89.52**	87.28	81.62	84.36
Localization	**83.21**	59.69	**69.51**	80.28	59.69	68.47
Binding	**65.36**	**40.73**	**50.19**	53.16	37.68	44.10
Complex events	**58.54**	41.14	**48.32**	55.73	**41.73**	47.72
Regulation	**62.22**	36.36	**45.90**	53.61	**36.62**	43.52
Positive regulation	**60.14**	**41.93**	**49.41**	57.90	41.37	48.26
Negative regulation	**53.19**	42.38	47.17	52.39	**46.06**	**49.02**
All events	**69.72**	**53.00**	**60.22**	67.01	52.14	58.65

Table 12: Detailed per-event performance of BEESL and KBTL (KB-driven TreeLSTM) on the test set.

Upper bound of the encoding We quantified the upper bound of our encoding strategy by directly evaluating the performance of the encoded development set. Results (P: 95.76%, R: 91.30%, F1: 93.48%) indicate the goodness of our strategy, and that the $\approx 6\%$ missing is due to cross-sentence arguments we disregard, similarly to previous work.

[6] http://bionlp-st.dbcls.jp/GE/2011/downloads/