Conundrums in Event Coreference Resolution: Making Sense of the State of the Art

Jing Lu and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{ljwinnie,vince}@hlt.utdallas.edu

Abstract

Despite recent promising results achieved by span-based approaches to event coreference resolution, there is a lack of understanding of what has been improved. We present an empirical analysis of our state-of-the-art span-based event coreference resolver (Lu and Ng, 2021) with the goal of providing the general NLP audience with a better understanding of the state of the art and coreference researchers with directions for future research.

1 Introduction

Recent years have seen the successful application of *span-based* neural models to key entity-based information extraction (IE) tasks such as entity coreference resolution (CR) (Lee et al., 2017, 2018) and relation extraction (Luan et al., 2019). Unlike many non-span-based neural models, which typically learn task-specific contextualized word representations (Peters et al., 2018), span-based models are designed to learn task-specific representations of *text spans*. This potentially allows span-based models to create better representations of the entity mentions involved in entity-based IE tasks than their non-span-based counterparts as many entity mentions are multi-word expressions.

Can the successes of span-based models be extended to event CR? The vast majority of existing event coreference resolvers were developed in the pre-neural NLP era, focusing primarily on feature engineering (Ahn, 2006; Chen et al., 2009; Chen and Ji, 2009; McConky et al., 2012; Araki et al., 2014; Bejan and Harabagiu, 2010, 2014; Chen and Ng, 2014, 2015, 2016; Cybulska and Vossen, 2015a,b; Yang et al., 2015; Krause et al., 2016; Lu et al., 2016) and adapting the models originally developed for entity coreference to event coreference (Liu et al., 2014; Peng et al., 2016; Lu and Ng, 2016, 2017, 2020). Neural event coreference models (Choubey and Huang, 2017, 2018, 2021; Huang et al., 2019) are few and far between, let alone span-based neural event coreference models.

Recently, Lu et al. (2020) designed the first span-based model that has achieved state-of-the-art results on a standard event coreference dataset. Despite these promising results, the use of span-based models in event-based IE tasks such as event coreference is still in its infancy. In particular, there is little understanding of what has been improved.

In light of the above discussion, we present an empirical analysis of our state-of-the-art span-based event coreference resolver (Lu and Ng, 2021) with the goal of gaining insights into its behavior. We believe that our analysis will not only provide the general NLP audience with a better understanding of the strengths and weaknesses of span-based event coreference models, but also provide coreference researchers with directions for future work.[1]

2 Tasks and Definitions

In this section, we define the six tasks to be learned by our span-based event coreference model.

The event coreference task involves identifying the event mentions in a document that refer to the same real-world event. In the example in Table 1, $ev1$ and $ev2$ are coreferent because they both refer to Ahrendts' starting to work for Apple. An event mention is composed of a *trigger*, a set of *arguments*, and a set of *attributes*, as defined below.

The trigger detection (TD) task aims to (1) extract from a document the event triggers, each of which is a word/phrase that expresses the occurrence of an event, and (2) assign an event subtype to each trigger that is chosen from a corpus-specific subtype inventory. In our example, $ev1$, $ev2$, and $ev3$ are triggered by "hire", "start", and "hired" respectively with subtype PERSONNEL_STARTPOSITION. Two mentions cannot be coreferent unless they have the same subtype.

[1] For an empirical analysis of non-neural event coreference models, we refer the reader to Chen and Ng (2013).

{Apple}$_{en1}$ said {Tuesday}$_{en2}$ that {it}$_{en3}$ will {hire}$_{ev1}$ {Angela Ahrendts, the chief executive of Burberry}$_{en4}$, as a member of {its}$_{en5}$ executive team. {She}$_{en6}$ will {start}$_{ev2}$ working for {Apple}$_{en7}$ in the {spring}$_{en8}$. In the {summer}$_{en9}$, {the company}$_{en10}$ {hired}$_{ev3}$ {Paul Deneve, the former CEO of Yves Saint Laurent}$_{en11}$, to work on special projects.

Table 1: Event coreference example.

	ACE			KBP		
	Train	Dev	Test	Train	Dev	Test
Number of documents	529	28	40	735	82	167
Number of gold event mentions	4202	450	403	20512	2382	4375
Number of event coreference chains	3272	313	290	13292	1502	2963
Average entropy of the subtype distribution of a gold event mention	0.122	0.047	0.059	0.414	0.217	0.268
Average entropy of the subtype distribution of a candidate event mention	0.299	0.403	0.433	0.313	0.368	0.517
Average entropy of the anaphoricity distribution of a gold event mention	0.370	0.264	0.275	0.485	0.335	0.375
Average entropy of the anaphoricity distribution of a candidate event mention	0.112	0.228	0.224	0.140	0.191	0.239

Table 2: Dataset statistics.

The argument extraction task aims to (1) extract the arguments of an event mention and (2) determine the role played by each argument in the event. In our example, $en3$ and $en4$ are the arguments of $ev1$ with roles ENTITY and PERSON respectively, for instance. Note that event arguments are entity mentions. Two event mentions cannot be coreferent if there is a role for which their arguments refer to different entities. For instance, $ev1$ and $ev3$ are not coreferent because the PERSON role of these two event mentions are filled by different entity mentions ("Angela Ahrendts" and "Paul Deneve").

The attribute prediction task aims to predict an event mention's attributes, each of which denotes a linguistic property of the event mention. For instance, one of the attributes defined in the ACE 2005 event coreference annotation guidelines is TENSE, which denotes the tense associated with an event and has four possible values, PAST, PRESENT, FUTURE, and UNSPECIFIED.

The anaphoricity prediction task determines whether an event mention is coreferent with any preceding mentions. For example, $ev1$ is non-anaphoric while $ev2$ is anaphoric.

Entity CR involves identifying the entity mentions in a document that refer to the same real-world entity. For instance, $en4$ and $en6$ are coreferent because they refer to Ahrendts.

3 Evaluation Setup

3.1 Datasets

We report results on two event coreference datasets that are the most comprehensively annotated, ACE 2005 and KBP 2017.[2] ACE 2005 defines 33 event subtypes, 30 argument roles, and four event attributes, whereas KBP 2017 defines 18 subtypes, 20 argument roles, and one attribute. For ACE 2005, while the official training set is available, the official test set is not. As a result, previous work defined different train-test partitions over the official training set when evaluating on ACE 2005. We employ the same train-test partition as Wadden et al. (2019). For the experiments involving KBP, we use five corpora (LDC2015E29, E68, E73, E94, and LDC2016E64) as our training set. Among them, we reserve 82 documents for parameter tuning. For evaluation, we use the official KBP 2017 test set.

Statistics on ACE 2005 and KBP 2017 are shown in Table 2.[3] In addition to general statistics such as the number of event coreference chains, we compute several statistics that aim to better gauge the difficulty of these datasets. The first one is the average entropy of the subtype distribution of an event mention. This statistic could shed light on the difficulty of TD: in general, the lower the entropy is, the easier it is to predict its subtype. We compute two versions of this statistic, one using gold mentions and the other using candidate mentions (i.e., gold mentions plus non-gold mentions created from words/phrases that have appeared as a trigger at least once in the dataset). When computing the entropy of the subtype distribution of a candidate mention, we include NONE as one of the subtypes. The second one is the average entropy of the anaphoricity distribution of a mention. This statistic could shed light on the difficulty of event CR: in general, the lower the entropy is, the easier it is to determine whether a mention is anaphoric. We compute it using both gold and candidate mentions.

[2] We excluded ECB+, an extensively used dataset, because of criticisms for its incomplete within-document event coreference annotation (Liu et al., 2014; Choubey and Huang, 2018).

[3] The event subtypes, argument roles, and event attributes defined for ACE 2005 and KBP 2017 can be found in LDC (2005) and LDC (2016), respectively.

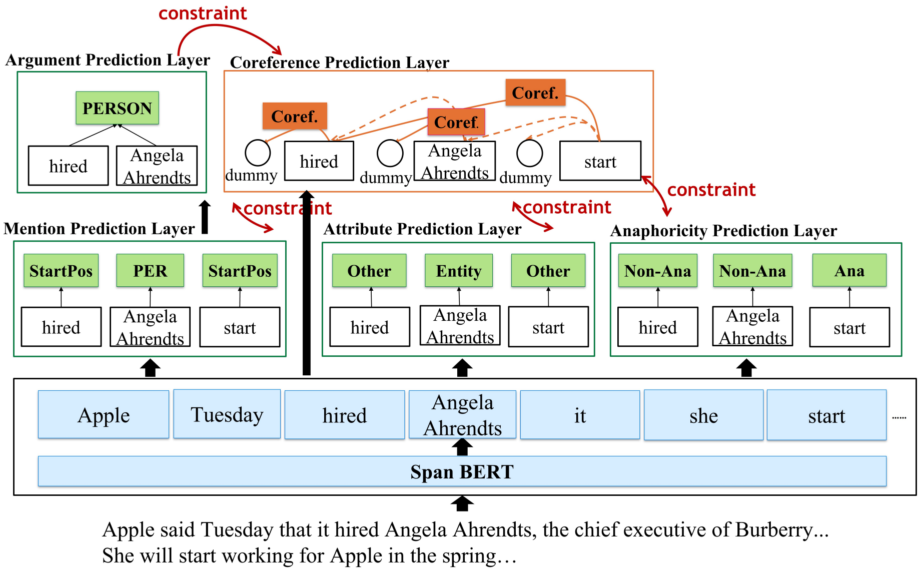

Figure 1: Model structure.

3.2 Evaluation Metrics

Results of event coreference are obtained using version 1.8 of the official scorer provided by the KBP 2017 shared task organizers. This scorer reports results in terms of AVG-F, which is the unweighted average of the F-scores of four coreference evaluation metrics, namely MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), CEAF$_e$ (Luo, 2005) and BLANC (Recasens and Hovy, 2011).

TD results are expressed in terms of F-score, where a trigger is considered correctly detected if it has an exact match with a gold trigger in terms of boundary and event subtype.

3.3 Model

Next, we provide an overview of our state-of-the-art span-based event coreference model.

Our model, which jointly learns six tasks, takes as input a document and divides it into non overlapping regions, following previous span-based models (Joshi et al., 2019). The word sequence in each region serves as an input training sequence, from which we extract all possible intra-sentence spans of up to length L. We then pass the sequence into a pre-trained Transformer encoder to encode the tokens and their contexts, and create a span's representation based on the encoded tokens in the Span Representation Layer. To maintain computational tractability, we score each span and retain only the top-scoring spans for further processing.

For each top span, we pass it to (1) the Mention Prediction Layer to predict its trigger subtype (or NONE if it is not a trigger); (2) the Anaphoricity Prediction Layer to predict its anaphoricity value (i.e., ANAPHORIC or NON-ANAPHORIC); (3) the Attribute Prediction Layer, which contains a network for predicting the value of each event attribute; (4) the Coreference Prediction Layer, which uses a ranker to link a span to its highest-ranked candidate antecedent (or NULL if the span is non-anaphoric). We learn entity and event coreference simultaneously by viewing them as a single coreference task. From a learning perspective, there is only one task to be learned, which is coreference resolution over a set of mentions. To do so, we extend the Span Representation Layer, the Mention Prediction Layer, and the Coreference Prediction Layer so that the mentions they identify/handle are composed of both entity and event mentions. Finally, for each span ev predicted to be an event mention and each span em predicted to be an entity mention that appears in the same sentence as ev, the Argument Prediction Layer predicts em's role in ev (or NULL if it is not an argument of ev). To guide the learning process, seven cross-task consistency constraints are enforced as *soft* constraints during both training and inference.

Figure 1 shows the structure of our model. For details, we refer the reader to Lu and Ng (2021).

We evaluate several variants of this model[4]:

SpanBERT-base vs. SpanBERT-large. To determine the impact of the encoder on span-based event coreference, we experiment with two encoders, SpanBERT-base (henceforth SpanBERT-b) and SpanBERT-large (henceforth SpanBERT-l),

[4] Details of parameter tuning can be found in the Appendix.

	ACE								KBP							
	Knowledge-Rich				Knowledge-Lean				Knowledge-Rich				Knowledge-Lean			
System	AvgF	MUC	Sing.	TD	AvgF	MUC	Sing.	TD	AvgF	MUC	Sing.	TD	AvgF	MUC	Sing.	TD
Pipeline models																
SpanBERT-b	57.6	55.6	50.4	73.9	58.4	56.4	50.4	73.9	42.5	36.9	54.7	62.2	44.0	38.9	54.6	62.2
SpanBERT-l	57.8	53.9	51.5	75.0	59.3	55.4	51.5	75.0	44.9	38.6	53.1	63.8	46.2	40.8	53.0	63.8
Joint models																
SpanBERT-b	58.5	56.2	55.5	73.1	57.9	56.8	55.5	72.3	44.2	39.6	42.7	62.9	43.9	40.5	39.4	62.1
SpanBERT-l	60.1	58.0	55.6	74.6	58.3	54.1	58.2	74.3	48.0	45.2	45.2	64.5	44.5	40.9	40.1	62.0

Table 3: Results of different model variants on two coreference datasets.

the latter of which is more complex.

Knowledge-rich vs. Knowledge-lean Resolution. While our model learns six tasks, many existing resolvers only perform two tasks, trigger detection and event coreference. To understand the benefits of the additional tasks, we experiment with both the six-task (henceforth Knowledge-Rich) version and the two-task (henceforth Knowledge-Lean) version of our model, the latter of which is obtained by removing the layers from our model that correspond to the remaining four tasks.

Pipeline vs. Joint Models. Although pipeline models are prone to error propagation, the majority of the existing resolvers are pipeline-based, performing trigger detection prior to event coreference. To evaluate the benefits of joint modeling, we evaluate a pipelined version of our model, where we first train a trigger detector, then use the resulting triggers to train an anaphoricity model and the attribute prediction models. Next, we train a joint entity extraction and entity coreference model, which is essentially the portion of our model containing only the Span Representation Layer, the Mention Prediction Layer, and the Coreference Prediction Layer. Then, we train an argument extraction model, which is the same as our model's Argument Prediction Layer, using the extracted entity mentions as candidate arguments for the triggers identified by the trigger detection model. Finally, the outputs of all these models are used to enforce the cross-task consistency constraints in our models as hard constraints, meaning that any candidate antecedent of an anaphor that violates a constraint is filtered prior to event CR.

4 Results of Model Variants

We first evaluate the variants of our model on ACE 2005 and KBP 2017. These variants differ along three dimensions, each of which has two choices: (1) SpanBERT-b vs. SpanBERT-l; (2) Knowledge-Rich vs. Knowledge-Lean; and (3) Pipeline vs. Joint. This yields eight possible variants, whose results are shown in Table 3.[5]

SpanBERT-b vs. SpanBERT-l. To study the impact of the encoder on event coreference performance, we compare the performances of the SpanBERT-b and SpanBERT-l resolvers. Event coreference results, which are expressed in AVG-F, are shown in the AvgF columns of Table 3. Keeping the other two dimensions fixed, we can see that a SpanBERT-l resolver always outperforms its SpanBERT-b counterpart w.r.t. event CR.

To better understand whether the SpanBERT-l resolvers offer better event coreference performance than their SpanBERT-b counterparts because of better TD, better identification of coreference links, better identification of singleton clusters, or a combination of these three factors, we report in Table 3 the TD F-scores in the TD columns, the MUC F-scores (which solely measure link prediction performance) in the MUC columns, and the percentage of singletons successfully recalled in the Sing. columns. We can see that while neither of them produces consistently better link identification results than the other, the SpanBERT-l resolvers rarely perform worse and sometimes perform substantially better than their SpanBERT-b counterparts w.r.t. TD and singleton identification. Nevertheless, despite the consistent improvement of SpanBERT-l over SpanBERT-b, the factors that contribute the most to SpanBERT-l's superior event coreference performance are different in different cases.

Knowledge-rich vs. Knowledge-lean. Would knowledge-rich event coreference models always outperform their knowledge-lean counterparts? The results in Table 3 reveal an interesting correlation: knowledge-rich models always perform worse than knowledge-lean models in the pipeline setting, whereas the reverse is true in the joint setting. Specifically, in the pipeline setting, knowledge-rich

[5] Owing to space limitations, we show only the most important scores in Table 3. The detailed results (e.g., B^3 and $CEAF_e$ results) can be found in the Appendix.

models underperform their knowledge-lean counterparts w.r.t. link identification. We speculate that the larger number of tasks trained in the knowledge-rich setting has aggravated error propagation, but additional experiments are needed to determine the reason. In contrast, in a joint setting, improvements in the knowledge-rich models are always accompanied by improvements in TD and sometimes by improvements in link identification and/or singleton identification.

Pipeline vs. Joint. While there have been claims that joint models of event coreference outperform their pipeline counterparts because of their ability to address error propagation, to our knowledge there has never been a head-to-head comparison of pipeline and joint models in a controlled setting that differ *only* w.r.t. whether the tasks involved are learned jointly or independently. The results in Table 3 reveal an interesting observation: joint models substantially outperform their pipeline counterparts in a knowledge-rich setting, whereas the reverse is true in a knowledge-lean setting. More specifically, we see that in a knowledge-rich setting, improvements in a joint model's coreference performance are always accompanied by significantly better link identification; in contrast, in a knowledge-lean setting, drops in a joint model's coreference performance can be attributed mostly to poorer performances on TD. Overall, these results provide suggestive evidence that there are indeed benefits in joint modeling when multiple tasks are involved.

ACE vs. KBP. Both the trigger detection results and the event coreference results on ACE are better than those on KBP. The reason can be attributed largely to the fact that the ACE test set is simpler than the KBP test set w.r.t. both trigger detection and event coreference, as explained below.

To understand why trigger detection is easier on ACE than on KBP, for each *candidate* event mention in the ACE/KBP test set, we computed the entropy of the event subtype distribution of its underlying trigger, and found that the entropy (averaged over the candidate mentions in the test set) is lower for ACE (0.433) than for KBP (0.517). This means that the subtype distribution of a candidate mention that appears in the ACE test set is on average more skewed than that in the KBP test set. At the same time, we found that our model's trigger detector was not good at exploiting context for subtype prediction, as most of the time the model simply assigned an event mention the subtype that co-occurred most frequently with the underlying trigger in the training set. Specifically, the average entropies of the *predicted* subtype distribution of a candidate mention in ACE and KBP are 0.330 and 0.366 respectively, which are lower than the average entropies of the *gold* subtype distribution mentioned above. Consequently, trigger detection results on KBP are worse than those on ACE.

The better event CR results obtained on ACE could in part be attributed to better TD, but the question is: is event CR easier on ACE than on KBP? To answer this question, we feed our model with gold event mentions (i.e., gold mention boundaries *and* gold subtypes) during testing. This means that any error observed in the output of the model must be due to incorrect resolution. The results are as follows. The percentage of gold event mentions in the test set that are resolved incorrectly is 11.3% for ACE and 19.2% for KBP. In other words, resolution performance on ACE is indeed better than that on KBP. These results suggest that the better coreference results on ACE can be attributed to not only better TD but also better resolution. The next question is: what makes event CR easier on ACE than on KBP? One reason is anaphoricity determination: the anaphoricity determination accuracies on the ACE and KBP test sets are 66.5% and 57.2% respectively. We believe that anaphoricity determination is easier on ACE because the event mentions are less ambiguous w.r.t. anaphoricity than those on KBP, as seen in the entropy values of anaphoricity distribution in Table 2.

5 Results on Resolution Classes

To gain additional insights into our best resolver (Joint Knowledge-Rich SpanBERT-l), we analyze its performance on different classes of event mentions by partitioning the *gold* event mentions in the test set into 11 *resolution classes*.

The first three classes contain event mentions that can be resolved via lemma matching or rote learning. **(1) Lemma match and seen pair (LM&SP).** An event mention e is assigned to this class if it has an antecedent (i.e., an event mention preceding e that is coreferent with e) such that the two have the same lemma *and* have appeared as a coreferent pair in the training data at least once. **(2) Lemma match only (LM).** An event mention is assigned to this class if it has an antecedent such that the two have the same lemma but have never appeared as a coreferent pair in the training data. **(3) Seen Pair (SP).** An event mention is assigned to this class if it has an antecedent such that the two have appeared as a coreferent pair in the training data at least once but do not have the same lemma.

We partition the remaining anaphoric mentions (i.e., those not covered by the first three classes) based on two factors. The first factor is the sentence distance (SD) between the gold mention e and its closest antecedent. The possible values are ≤ 1 (if e's antecedent appears in either the same sentence as e or the preceding sentence) and > 1 (otherwise). The second factor depends on the number of coreferent arguments (CA) shared by e and an antecedent. The possible values are > 1 (if e has an antecedent a such that when comparing the event arguments of e and a w.r.t. each role, there are at least two roles containing coreferent arguments), $= 1$ (if e fails the > 1 condition, but has an antecedent such that the two have exactly one role in which their arguments are coreferent) and $= 0$ (otherwise). We use these two factors to define six classes: **(4) SD** ≤ 1**, CA** > 1, **(5) SD** ≤ 1**, CA** $= 1$, **(6) SD** ≤ 1**, CA** $= 0$, **(7) SD** > 1**, CA** > 1, **(8) SD** > 1**, CA** $= 1$, and (9) **SD** > 1**, CA** $= 0$.

Finally, we partition the non-anaphoric event mentions into two classes: **(10) Seen non-anaphoric (SNA) mentions.** A non-anaphoric mention is assigned to this class if it is seen in the training set. **(11) Unseen non-anaphoric (UNA) mentions.** A non-anaphoric mention is assigned to this class if it does not appear in the training set.

Results of our resolver on these 11 resolution classes are shown in Table 4. Specifically, for each resolution class C, we show its TD recall (percentage of gold mentions in C that are correctly recalled by the trigger detector) under **TD** and its resolution accuracy (percentage of correctly identified anaphors in C that are correctly resolved)[6] under **RA**. Under **Size** we show the percentage of gold mentions belonging to each resolution class.

		ACE			**KBP**		
	Class	Size	RA	TD	Size	RA	TD
1	LM&SP	12.9	72.5	98.1	6.3	83.3	72.7
2	LM	3.0	28.6	58.3	8.4	77.0	59.4
3	SP	4.7	72.2	94.7	1.7	54.9	72.9
4	SD≤1, CA>1	0.7	0.0	66.7	1.4	41.5	71.9
5	SD≤1, CA=1	0.3	0.0	0.0	2.4	50.0	62.6
6	SD≤1, CA=0	1.2	33.3	60.0	2.8	35.2	46.6
7	SD>1, CA>1	1.7	60.0	71.4	1.8	50.0	65.8
8	SD>1, CA=1	1.7	33.3	85.7	4.6	51.3	61.6
9	SD>1, CA=0	1.8	60.0	71.4	3.5	42.2	43.8
10	SNA	64.5	87.4	85.4	60.0	88.3	72.8
11	UNA	7.4	100.0	40.0	7.2	89.8	29.3

Table 4: Results on resolution classes.

Among the resolution classes involving anaphoric mentions, the ones with the highest RAs are LM&SP, LM, and SP. This should not be surprising: lemma matching and memorization of pairs seen in the training set are by far among the most reliable indicators of event coreference. TD F-scores are lower for LM than for the other two classes. This should not be surprising either, as TD performance is strongly correlated with whether the mentions are seen in the training set or not.

The six classes defined by SD and CA contain anaphors that are harder to resolve than the above three owing to the absence of reliable indicators of event coreference. Two points deserve mention. First, somewhat contrary to expectation, sentence distance does not seem to affect resolution difficulty, as the RAs of the SD≤1 classes are not higher than those of the SD>1 classes. This may have to do with the fact that these RAs are only computed over difficult resolution classes. Second, on KBP, we see that the RAs of the CA>1 and CA=1 classes are higher than the RA of the CA=0 class. These results seem to be consistent with our intuition that an event mention is easier to resolve if it shares common arguments with its antecedent. Interestingly, the opposite seems to be true on ACE. This may have to do with the fact that the number of gold event mentions in the ACE test set is much smaller than that in the KBP test (403 vs. 4174). In particular, the number of anaphors covered by the six classes involving SD and CA in ACE is probably too small to draw reliable conclusions.

The highest RAs are achieved by the two non-anaphoric resolution classes, SNA and UNA. For non-anaphoric event mentions, a high RA implies that the corresponding resolver has successfully determined that these mentions should not be resolved. In addition, the TD F-scores associated with these two classes again confirm that TD performance correlates with the percentage of mentions that are seen in the training set.

6 Sensitivity to Perturbed Inputs

Next, we conduct a series of experiments that involve perturbing the input. In each experiment, we (1) replace a certain kind of words/phrases in each training document with other words/phrases, (2)

[6] In other words, the resolution accuracy does not depend on anaphor recall and precision, as it is computed only over those mentions that are successfully recalled by the resolver.

train a coreference model on these perturbed training documents, and (3) evaluate the output. Our goal is to gain insights into the behavior of our span-based resolver by examining how sensitive its performance is to perturbations in the input. Specifically, if performance drops significantly when a particular kind of words/phrases is replaced, that means the replaced words/phrases are important in the model learning process. Note that perturbations are only applied to the training documents: no changes are made to the test documents.

We divide the different kinds of perturbations into two broad categories, mention-internal perturbations and mention-external perturbations.

6.1 Mention-internal Perturbations

Mention-internal perturbations involve making changes to the words associated with an event mention (e.g., its trigger, its arguments).

6.1.1 Perturbation to Event Triggers

We replace each trigger, t_1, in a training document with another trigger, t_2, that appears in the training set. This ensures that the number of triggers that are unseen w.r.t. the training set will not change. Importantly, the replacement is deterministic, meaning that (1) all occurrences of t_1 will be replaced with the same trigger (i.e., t_2), and (2) any trigger coreferent with t_1 (but are not lexically identical to it, such as "died" and "passed away") will be replaced with a trigger that has been coreferent with t_2 at least once in the training data. These conditions ensure that only the triggers will change, but their event coreference relationships will not. Due to the randomness involved in the choice of t_2, we repeat the experiment three times and report the average result.

6.2 Perturbation to Event Arguments

We have two kinds of perturbations to arguments.

Arguments with entity coreference relations preserved. This experiment is the same as the trigger perturbation experiment described in the previous subsection except that we are now replacing arguments rather than triggers.

Arguments with entity coreference relations disrupted. This experiment is the same as the "arguments with entity coreference relations preserved" experiment, except that entity coreference relations are disrupted in the replacement process. More specifically, while the replacement of an argument a_1 with another argument a_2 has so far been deterministic, in this experiment the replacement is random, meaning that different occurrences of a_1 may be replaced with different arguments. Moreover, to ensure that entity coreference chains are disrupted, we ensure that if a_1 is replaced with a_2, then any entity mention coreferent with a_1 will be replaced with another entity mention that is not coreferent with a_2. This experiment will shed light on the impact of entity coreference relations in the model learning process.

6.3 Mention-external Perturbations

Mention-external perturbations involve making changes to the words outside an event mention.

6.3.1 Perturbation to Entity Mentions

We replace each entity mention e_1 that is not an event argument with another entity mention of the same entity type. The replacement is random, meaning that different occurrences of e_1 may be replaced with different entity mentions. Unlike the argument-related experiments, we do not make any attempt to explicitly preserve or disrupt the entity coreference relations in the replacement process.

6.3.2 Perturbation to Verbs

We replace each verb in the surrounding context with a different verb that is taken from the training set but has never appeared as a trigger in the training set. This replacement strategy ensures that (1) the number of verbs that are unseen w.r.t. the training set will not change, and (2) the model will not be misguided when learning from the verbs (i.e., a verb that is unambiguously used as a trigger will not be misled as an ambiguous verb because of the replacement strategy). Note that the replacement is deterministic: all occurrences of a given verb will be replaced with the same verb. To avoid confusing the learner, the new verb should never appear as a trigger in the training set.

6.3.3 Perturbation to Adjectives & Adverbs

This experiment is the same as the "perturbations to verbs" experiment, except that we replace adjectives and adverbs rather than verbs.

6.4 Perturbation Results

Results of these experiments, which are shown in Table 5, are expressed in terms of coreference resolution AVG-F score (in the CR columns) and TD F-score (in the TD columns). To facilitate comparison, we show in row 1 the performance of our resolver when the input is not perturbed.

		ACE		KBP	
	Perturbation Type	CR	TD	CR	TD
1	No perturbation	60.1	74.6	48.0	64.5
2	Trigger	57.6	73.5	43.3	62.1
3	Arg:Coref Preserved	52.5	68.3	46.4	63.2
4	Arg:Coref Disrupted	52.0	68.2	45.9	64.1
5	Entity mentions	56.4	72.4	46.3	64.0
6	Verbs	57.1	72.5	47.0	64.4
7	Adjectives/Adverbs	57.9	73.9	46.7	64.5

Table 5: Perturbation results.

Several points deserve mention. First, all CR and TD results obtained via perturbations are lower than the "No perturbation" results in row 1. This implies that each kind of perturbation we considered affects the model learning process and negatively impacts event CR and TD performances. Second, while intuitively the mention-internal perturbations, particularly the perturbation involving triggers, should negatively impact event CR performance more than their mention-external counterparts, it is interesting to see that this intuition is only somewhat supported by the ACE results and certainly not by the KBP results. In fact, on KBP there is not a perturbation that is obviously more disruptive than the others. Even more interesting are the TD results: while intuitively the perturbation on triggers would be more disruptive to TD performance, our results suggest that this is not the case. Overall, these results suggest that while our state-of-the-art event CR resolver is sensitive to every kind of perturbations we experimented with, it is not particularly more sensitive to any one of them.

7 Using Oracles

How can the performance of our best resolver (Joint Knowledge-Rich SpanBERT-l) be improved? To answer this question, we perform oracle experiments in which we feed the resolver with different types of perfect information and examine how much performance gains can be obtained.

7.1 Gold Mention Boundaries

Our first oracle experiment concerns training and testing our resolver on gold mention boundaries. This experiment will enable us to determine the extent to which event coreference performance can be improved if we improve span (boundary) detection. Specifically, we disable the component in our resolver that is responsible for proposing spans (i.e., mention boundaries) and instruct it to use gold mention spans instead. Note, however, that the representations of a span will still be learned during training and then used during testing.

	Oracle	ACE	KBP
1	None	60.1	48.0
2	Gold boundaries	77.1	62.9
3	Gold anaphoricity	65.9	52.7
4	Gold subtypes	78.0	69.2
5	Gold boundaries & subtypes	83.3	80.1
6	Subtype agreement	62.5	48.6
7	Arg+EC agreement	60.8	48.1
8	Realis agreement	N/A	49.8
9	Modality agreement	62.3	N/A
10	Tense agreement	64.0	N/A
11	Genericity agreement	63.0	N/A
12	Polarity agreement	63.0	N/A

Table 6: Results of the oracle experiments.

Results, which are expressed in terms of AVG-F, are shown in row 2 of Table 6. For convenience, we show the results of our best model variant in row 1. Comparing these two rows, not only do we see consistent improvements across the two datasets, but the improvements are substantial: AVG-F can be improved by as much as 14.9–17.0% points.

7.2 Gold Anaphoricity

In our second oracle experiment, we aim to shed light on the role anaphoricity plays in event coreference by providing our resolver with perfect event anaphoricity information, meaning that we know for every event mention whether it is anaphoric or not. We use this perfect anaphoricity information during resolution: we will resolve all and only those event mentions that are anaphoric.

Results are shown in row 3 of Table 6. As we can see, our resolver improved on both datasets (by 4.7–5.8% points in AVG-F score). These results imply that further improvements in anaphoricity can improve event CR.

7.3 Gold Subtypes

In our third oracle experiment, we assume that our resolver is given gold event subtypes. Specifically, we use gold subtypes in lieu of predicted subtypes whenever the latter is needed by the model during testing. In addition, we use gold subtypes in (1) *resolution*, where we disallow coreference between spans with different gold subtypes during testing, and (2) *postprocessing*, in which we remove all spans whose gold subtypes are NULL from the system output prior to scoring.

Results are shown in row 4 of Table 6. The improvements obtained via gold event subtypes are even more substantial than those obtained via gold mention boundaries and perfect anaphoricity:

AVG-F scores increase by 17.9–21.2% points. We believe that the reasons are two-fold. One has to do with the removal of spans whose gold subtype is NULL during postprocessing: this effectively brings the precision of TD to 100%. The other has to do with how scoring is done in event coreference: the official scorer considers a singleton/non-singleton cluster correctly identified if and only if the subtype predicted for each event mention in the cluster is correct. In other words, event coreference performance depends heavily on accurate event subtyping. Overall, these results suggest that we can go a long way in event coreference by focusing on improving event subtyping.

7.4 Gold Mention Boundaries and Subtypes

In our fourth oracle experiment, we assume that our resolver is given gold mention boundaries and event subtypes. In essence, we are using the two oracles described in Sections 7.1 and 7.4 in combination. Together they ensure that the recall and the precision of TD are both 100%. Given this perfect TD component, any errors made by the resolver can be attributed solely to resolution.

Results are shown in row 5 of Table 6. As we can see, employing gold boundaries and gold event subtypes in combination yields substantially better results than employing these oracles in isolation. Specifically, the use of gold subtypes on top of gold boundaries yields an improvement of 6.2–17.2% points in AVG-F score, whereas the use of gold boundaries on top of gold subtypes yields an improvement of 5.3–10.9% points in AVG-F score. The AVG-F scores on the two datasets both exceed 80%: these scores represent the upper bound on event coreference performance that can be achieved by our resolver solely by improving TD.

7.5 Agreement Oracles

Our next set of experiments involves the use of *agreement oracles*. Each oracle takes two event mentions as input and returns a *binary* value that indicates whether they satisfy a certain condition that encodes a linguistic constraint on coreference. Our resolver uses an agreement oracle during resolution: if the oracle says that the condition being tested is not satisfied between the two event mentions, the resolver will not establish any coreference link between them.

Our first agreement oracle is the event subtype oracle, which checks whether two event mentions have the same gold subtype. Note that this oracle is different from the oracle described in Section 7.3: this oracle only returns a binary value indicating whether the given event mentions have the same gold subtype, but unlike the previous oracle, it does not tell us what their gold subtypes are. Our second agreement oracle is the argument+entity coreference oracle, which checks whether two event mentions have a role in which the two arguments are not entity-coreferent. The remaining agreement oracles check whether the two event mentions agree w.r.t. one of its event attributes, namely MODALITY, TENSE, GENERICITY and POLARITY for ACE, and REALIS for KBP.

Results are shown in rows 6–12 of Table 6. As we can see, all agreement oracles are only mildly useful. This is understandable: since these oracles are used to disallow coreference between two mentions that violate a certain linguistic constraint, they can help improve coreference precision but not recall. Among these oracles, those involving the event attributes tend to be more useful the remaining two, which involve subtypes (row 6) and arguments+entity coreference (row 7). In other words, merely knowing whether two mentions agree in event subtype and whether they have non-coreferent arguments in their respective roles offer little help in improving coreference performance.

8 Conclusion

While space limitations preclude a reiteration of all the observations we made in our empirical analysis of our resolver, we believe the key conclusions are: (1) a knowledge-rich joint event CR model trained using SpanBERT-l achieves better performance than other model variants; (2) resolving anaphoric event mentions that cannot be resolved using lemma matching and memorization of mention pairs seen in the training set remains a challenge in event CR; (3) while our state-of-the-art span-based resolver is sensitive to all kinds of perturbations we considered, there is not one perturbation it is particularly sensitive to; and (4) improving mention boundary detection, anaphoricity detection, and subtype detection will likely lead to the largest improvement in event CR performance.

Acknowledgments

We thank the three anonymous reviewers for their helpful comments. The paper's title was motivated by that of Stoyanov et al.'s (2009) paper. This work was supported in part by NSF Grant IIS-1528037.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

Jun Araki, Zhengzhong Liu, Eduard Hovy, and Teruko Mitamura. 2014. Detecting subevent structure for event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the LREC Workshop on Linguistic Coreference*, pages 563–566.

Cosmin Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1412–1422, Uppsala, Sweden. Association for Computational Linguistics.

Cosmin Adrian Bejan and Sanda Harabagiu. 2014. Unsupervised event coreference resolution. *Computational Linguistics*, 40(2):311–347.

Chen Chen and Vincent Ng. 2013. Chinese event coreference resolution: Understanding the state of the art. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 822–828, Nagoya, Japan. Asian Federation of Natural Language Processing.

Chen Chen and Vincent Ng. 2014. SinoCoreferencer: An end-to-end Chinese event coreference resolver. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4532–4538, Reykjavik, Iceland. European Language Resources Association (ELRA).

Chen Chen and Vincent Ng. 2015. Chinese event coreference resolution: An unsupervised probabilistic model rivaling supervised resolvers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1097–1107, Denver, Colorado. Association for Computational Linguistics.

Chen Chen and Vincent Ng. 2016. Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2913–2920. AAAI Press.

Zheng Chen and Heng Ji. 2009. Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4)*, pages 54–57, Suntec, Singapore. Association for Computational Linguistics.

Zheng Chen, Heng Ji, and Robert Haralick. 2009. A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the Workshop on Events in Emerging Text Types*, pages 17–22, Borovets, Bulgaria. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, Copenhagen, Denmark. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2018. Improving event coreference resolution by modeling correlations between event coreference chains and document topic structures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 485–495, Melbourne, Australia. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2021. Automatic data acquisition for event coreference resolution. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1185–1196, Online. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2015a. "Bag of events" approach to event coreference resolution. Supervised classification of event templates. *International Journal of Computational Linguistics and Applications*, 6(2):11–27.

Agata Cybulska and Piek Vossen. 2015b. Translating granularity of event slots into features for event coreference resolution. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Yin Jou Huang, Jing Lu, Sadao Kurohashi, and Vincent Ng. 2019. Improving event coreference resolution by learning argument compatibility from unlabeled data. 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 785–795, Minneapolis, Minnesota. Association for Computational Linguistics.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. 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 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Sebastian Krause, Feiyu Xu, Hans Uszkoreit, and Dirk Weissenborn. 2016. Event linking with sentential features from convolutional neural networks. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 239–249, Berlin, Germany. Association for Computational Linguistics.

LDC. 2005. ACE (Automatic Content Extraction) English annotation guidelines for events Version 5.4.3. Linguistic Data Consortium.

LDC. 2016. Rich ERE annotation guidelines overview V4.2. Linguistic Data Consortium.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. Supervised within-document event coreference using information propagation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4539–4544, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jing Lu and Vincent Ng. 2016. Event coreference resolution with multi-pass sieves. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3996–4003, Portorož, Slovenia. European Language Resources Association (ELRA).

Jing Lu and Vincent Ng. 2017. Joint learning for event coreference resolution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–101, Vancouver, Canada. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2020. Event coreference resolution with non-local information. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 653–663, Suzhou, China. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2021. Constrained multi-task learning for event coreference resolution. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4504–4514, Online. Association for Computational Linguistics.

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint inference for event coreference resolution. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3264–3275, Osaka, Japan. The COLING 2016 Organizing Committee.

Yaojie Lu, Hongyu Lin, Jialong Tang, Xianpei Han, and Le Sun. 2020. End-to-end neural event coreference resolution. *CoRR*, abs/2009.08153.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. 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 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Katie McConky, Rakesh Nagi, Moises Sudit, and William Hughes. 2012. Improving event coreference by context extraction and dynamic feature weighting. In *CogSIMA*, pages 38–43.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. 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.

Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand Index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 656–664, Suntec, Singapore. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. 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 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:517–528.

A Hyperparameter Tuning

For each coreference resolver, we have ten hyperparameters to tune. Specifically, we search for (1) the max span width (i.e., the maximum number of words in a candidate span) out of {5, 10, 15 }; (2) the max top antecedents (i.e., the maximum number of candidate antecedents) out of [10, 50] with step size 5; (3) the max training segments out of {3, 4, 5}; (4) the top span ratio (i.e., the fraction of top spans that survive the filtering) out of {0.3, 0.4, 0.5}; (5) the max segment length out of {128, 256, 384, 512}; (6) the task learning rate out of {5e-6, 1e-5, 1e-4}; (7) the weights associated with the three types of mistakes made by the coreference model: α_{WL} (i.e., the number of incorrectly resolved anaphoric mentions), α_{FN} (i.e., the number of non-anaphoric mentions misclassified as anaphoric), and α_{FA} (i.e., the number of anaphoric mentions misclassified as non-anaphoric) out of { 0.1, 0.2, 0.5, 1, 2, 5, 10 }; (8) the weights associated with the three types of mistake made by the mention prediction model: $\alpha_{t,WL}$ (i.e., the number of mentions labeled with the wrong subtype), $\alpha_{t,FN}$ (i.e., the number of non-mentions misclassified as mentions), and $\alpha_{t,FT}$ (i.e., the number of mentions misclassified as non-mentions) out of { 0.1, 0.2, 0.5, 1, 2, 5, 10 }; (9) the weights associated with attribute prediction, which are defined in a similar manner to those used in trigger detection; and (10) the weights associated with argument prediction, which are also defined in a similar manner to those used in trigger detection. We set the SpanBERT learning rate to 2e-5. Table 7 shows the best hyperparameter setting of each model variant on ACE

	Pipeline				Joint			
	Knowledge-Lean		Knowledge-Rich		Knowledge-Lean		Knowledge-Rich	
Hyperparameter	SpanB-b	SpanB-l	SpanB-b	SpanB-l	SpanB-b	SpanB-l	SpanB-b	SpanB-l
ACE								
Max span width	10	10	10	10	5	5	5	5
Max top antecedents	35	35	35	35	10	10	10	10
Max training segments	5	3	5	3	5	3	5	3
Top span ratio	0.4	0.4	0.4	0.4	0.3	0.3	0.3	0.3
Max segment length	384	512	384	512	384	512	384	512
Task learning rate	1e-5	5e-6	1e-5	5e-6	1e-4	1e-4	1e-4	1e-4
(α_{WL},α_{FN},α_{FA})	(1,.1,1)	(.5,.1,1)	(.5,.1,5)	(1,1,.5)	(.1,.1,1)	(.5,.2.5)	(.5,.1,1)	(.5,.1,5)
($\alpha_{t,WL}$,$\alpha_{t,FN}$,$\alpha_{t,FT}$)	(.2,.1,1)	(.5,.1,5)	(.2,.1,1)	(.5,.1,5)	(.1,.1,1)	(.1,.5,5)	(.5,.5,5)	(.5,.5,10)
($\alpha_{a,WL}$,$\alpha_{a,FN}$,$\alpha_{a,FT}$)	–	–	(.2,.1,1)	(.5,.5,1)	–	–	(.5,.5,1)	(1,1,1)
($\alpha_{o,WL}$,$\alpha_{o,FN}$,$\alpha_{o,FT}$)	–	–	(1,1,1)	(1,1,1)	–	–	(1,1,1)	(1,1,1)
KBP								
Max span width	5	5	5	5	5	5	5	5
Max top antecedents	50	50	50	50	15	15	15	15
Max training segments	5	3	5	3	5	3	5	3
Top span ratio	0.4	0.4	0.4	0.4	0.5	0.5	0.5	0.5
Max segment length	384	512	384	512	384	512	384	512
Task learning rate	1e-5	1e-5	1e-5	1e-5	1e-4	1e-4	1e-4	1e-4
(α_{WL},α_{FN},α_{FA})	(.1,.1,1)	(1,1,1)	(1,1,1)	(.5,.1,1)	(1,.1,10)	(.1,.1,5)	(1,.1,3)	(1,.1,5)
($\alpha_{t,WL}$,$\alpha_{t,FN}$,$\alpha_{t,FT}$)	(.1,.1,1)	(.1,.1,5)	(.1,.1,1)	(.1,.1,5)	(.5,.1,5)	(.1,.1,5)	(.5,.1,3)	(.5,.1,3)
($\alpha_{a,WL}$,$\alpha_{a,FN}$,$\alpha_{a,FT}$)	–	–	(.1,.1,5)	(.1,.1,5)	–	–	(.5,.1,1)	(.5,.1,1)
($\alpha_{o,WL}$,$\alpha_{o,FN}$,$\alpha_{o,FT}$)	–	–	(.5,.5,1)	(1,1,1)	–	–	(.5,.5,1)	(.5,.5,1)

Table 7: Best hyperparameters obtained on the development set for each resolver.

	Coreference													TD		
	MUC			B^3			CEAF$_e$			BLANC			AVG			
	R	P	F	R	P	F	R	P	F	R	P	F	F	R	P	F
ACE																
Pipeline, Knowledge-Lean																
SpanBERT-b	56.6	56.1	56.4	64.1	66.4	65.2	59.8	63.6	61.7	50.9	49.7	50.1	58.4	72.4	75.5	73.9
SpanBERT-l	54.9	55.9	55.4	65.6	67.3	66.4	63.6	64.5	64.0	52.9	50.0	51.4	59.3	74.4	75.6	75.0
Pipeline, Knowledge-Rich																
SpanBERT-b	68.1	47.0	55.6	67.9	59.8	63.6	53.8	70.3	60.9	56.8	47.6	50.4	57.6	72.4	75.5	73.9
SpanBERT-l	58.4	50.0	53.9	66.8	63.4	65.1	58.6	64.2	61.3	55.2	48.1	51.1	57.8	74.4	75.6	75.0
Joint, Knowledge-Lean																
SpanBERT-b	55.8	57.8	56.8	69.8	60.2	64.6	69.0	53.4	60.2	58.1	44.5	50.1	57.9	79.5	66.3	72.3
SpanBERT-l	53.1	55.0	54.1	65.6	65.6	65.6	64.5	62.1	63.3	52.0	48.9	50.4	58.3	74.9	73.7	74.3
Joint, Knowledge-Rich																
SpanBERT-b	54.0	58.6	56.2	66.5	64.1	65.3	65.9	56.9	61.1	55.5	48.0	51.3	58.5	76.4	70.1	73.1
SpanBERT-l	61.1	55.2	58.0	71.9	61.6	66.4	65.9	57.8	61.6	62.8	48.2	54.5	60.1	79.5	70.3	74.6
KBP																
Pipeline, Knowledge-Lean																
SpanBERT-b	35.7	39.5	37.5	48.0	51.7	49.8	44.0	53.5	48.3	29.1	38.1	32.9	42.1	57.5	67.7	62.2
SpanBERT-l	34.3	50.2	40.8	46.8	61.0	53.0	46.9	59.1	52.3	29.5	45.6	35.6	45.4	56.1	73.9	63.8
Pipeline, Knowledge-Rich																
SpanBERT-b	33.9	40.5	36.9	47.2	53.8	50.3	46.4	54.3	50.0	28.2	39.5	32.9	42.5	57.5	67.7	62.2
SpanBERT-l	31.6	49.7	38.6	45.6	63.1	53.0	48.1	58.9	53.0	27.1	49.3	35.0	44.9	56.1	73.9	63.8
Joint, Knowledge-Lean																
SpanBERT-b	37.1	47.3	41.6	49.8	53.0	51.4	48.4	48.9	48.7	32.2	38.7	35.2	44.2	60.1	65.1	62.5
SpanBERT-l	31.8	57.3	40.9	43.9	64.0	52.1	45.6	56.9	50.6	26.2	50.5	34.5	44.5	53.3	73.9	62.0
Joint, Knowledge-Rich																
SpanBERT-b	32.9	49.9	39.6	47.4	57.2	51.9	49.8	50.7	50.3	29.8	42.5	34.9	44.2	67.2	59.0	62.9
SpanBERT-l	39.1	53.6	45.2	49.4	61.2	54.7	49.7	58.6	53.8	32.1	47.2	38.2	48.0	58.7	71.6	64.5

Table 8: Results of the resolvers according to different evaluation metrics on the two coreference datasets.

and KBP.

B Results from Different Evaluation Metrics

Table 8 shows the detailed event coreference results, which are expressed in terms of recall (R), precision (P) and F-score (F) that are obtained via different evaluation metrics (i.e., MUC, B^3, CEAF$_e$, and BLANC). In addition, we express trigger detection performance in terms of R, P, and F. As can be seen, the Joint Knowledge-Rich SpanBERT-large model outperforms other model variants w.r.t. all metrics for KBP and all metrics except CEAF$_e$ for ACE.