Neural Anaphora Resolution in Dialogue Revisited

Shengjie Li and **Hideo Kobayashi** and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{sxl180006,hideo,vince}@hlt.utdallas.edu

Abstract

We present the systems that we developed for all three tracks of the CODI-CRAC 2022 shared task, namely the anaphora resolution track, the bridging resolution track, and the discourse deixis resolution track. Combining an effective encoding of the input using the SpanBERT$_{\text{Large}}$ encoder with an extensive hyperparameter search process, our systems achieved the highest scores in all phases of all three tracks.

1 Introduction

Following the CODI-CRAC 2021 shared task (Khosla et al., 2021), the CODI-CRAC 2022 shared task (Yu et al., 2022) focuses on resolving anaphoric references in dialogue. The two shared tasks are structured in more or less the same way. Specifically, in order to track progress on resolving anaphoric references in dialogue made over the past year, this year's shared task has essentially the same format as last year's, except that a new Gold Anaphor (Gold A) phase is added to both the bridging resolution track and the discourse deixis resolution track. By providing the participants with gold anaphors, the Gold A phase allows participants to focus on antecedent selection, thus making it easier to compare different systems' resolution performances.

Similar to last year, this year we participated and ranked first in all phases of all three tracks. We believe that our success can largely be attributed to two factors. First, leveraging the successes achieved by span-based models in last year's shared task (Kobayashi et al., 2021; Xu and Choi, 2021), we employ SpanBERT$_{\text{Large}}$ (Joshi et al., 2020) as our encoder in all three tracks. Second, we combine the resulting effective encoding of the input documents with an extensive hyperparameter search process. More specifically:

- for anaphora resolution, we employ a three-step pipeline approach consisting of mention extraction, entity coreference resolution, and removal of non-referring and non-entity mentions, pretraining the mention extraction component and the entity coreference component on the OntoNotes 5.0 corpus;
- for discourse deixis resolution, we propose a number of task-specific extensions to the span-based model we used in last year's shared task (Kobayashi et al., 2021) that involve heuristically extracting candidate anaphors and antecedents, exploiting different types of features, and performing distance-based filtering of candidate antecedents;
- for bridging resolution, we extend Yu and Poesio's (2020) multi-task learning framework, which jointly identifies bridging and coreference links, by replacing its LSTM encoder with SpanBERT$_{\text{Large}}$ and employing a *turn* distance feature.

A brief overview of the approaches we adopted for the three tracks can be found in Table 1.

The rest of the paper is structured as follows. The next three sections describe our work for the three tracks, namely entity coreference (Section 2, discourse deixis (Section 3), and bridging (Section 4). In each section, we describe our approach, our official results, and a brief analysis of the results, particularly a discussion of the impact of hyperparameter tuning on model performance. Finally, we present our conclusions in Section 5.

2 Anaphora Resolution

Last year we built a span-based entity coreference model for the Anaphora Resolution track that achieved competitive performance (Kobayashi et al., 2021). Since this year's Anaphora Resolution track has the exact same format as last year's, we developed this year's model based on last year's model (henceforth `UTD`^{2021}). Recall that `UTD`^{2021} is an extension of Xu and Choi's (2020)

Entity Coreference Resolution	
Baseline	Kobayashi et al.'s (2021) implementation of Xu and Choi.'s (2020) span-based model
Learning framework	A pipeline architecture consisting of a mention detection component, an entity coreference component, and a non-entity and non-referring mention removal component. The coreference component extends the baseline by (1) removing the type prediction model; and (2) rescoring the dummy antecedent at inference time to adjust the likelihood it will be selected as the antecedent.
Markable extraction	A mention detection model (adapted from Kobayashi et al. (2021)) is trained to identify the entity mentions.
Training data	The first two steps of our pipelined approach are pretrained on OntoNotes 5.0. All three steps of our pipelined approach are trained on ARRAU 3.0 (RST, GNOME, TRAINS91, TRAINS93, PEAR, $\text{LIGHT}_{\text{train}}$, $\text{AMI}_{\text{train}}$, $\text{Persuasion}_{\text{train}}$, $\text{Switchboard}_{\text{train}}$).
Development data	For all three steps, $\text{LIGHT}_{\text{dev}}$, AMI_{dev}, $\text{Persuasion}_{\text{dev}}$, and $\text{Switchboard}_{\text{dev}}$ are used. Note that after parameters are tuned on the dev data, we retrain the models on the combined training and dev sets using the tuned parameters before continuing parameter tuning on the test data. See Section 2.4.3 for details.
Discourse Deixis Resolution	
Baseline	Xu and Choi's (2020) implementation of Lee et al.'s (2018) span-based model
Learning framework	An extension of Xu and Choi's model with (1) heuristic extraction of candidate anaphors and antecedents, (2) an anaphor prediction model with which only those spans predicted as anaphors will be resolved, (3) a large-scale expansion of statistical features, and (4) filtering of candidate antecedents based on their distances from the anaphor under consideration. The models developed for the three phases differ w.r.t. the candidate anaphors they are trained on: in the Predicted phase, the model is trained on heuristically extracted candidate anaphors; in the Gold Mention phase, the model is trained on gold mentions; and in the Gold Anaphor phase, the model is trained on gold anaphors with gold mentions as their candidate antecedents.
Markable extraction	For the Predicted phase, markables are heuristically extracted. For the Gold Mention and Gold Anaphor phases, gold mentions and gold anaphors are used as candidate anaphors respectively. For all phases, candidate antecedents are extracted heuristically (utterances).
Training data	ARRAU 3.0 (RST, GNOME, TRAINS91, TRAINS93, PEAR, $\text{LIGHT}_{\text{train}}$, $\text{LIGHT}_{\text{dev}}$, $\text{AMI}_{\text{train}}$, AMI_{dev}, $\text{Persuasion}_{\text{train}}$, $\text{Persuasion}_{\text{dev}}$, $\text{Switchboard}_{\text{train}}$, $\text{Switchboard}_{\text{dev}}$).
Development data	None: we perform parameter tuning directly on the test data.
Bridging Resolution	
Baseline	Yu and Poesio's (2020) multi-task learning (MTL) framework
Learning framework	An extension of Yu and Poesio's framework in which we (1) replace their LSTM encoder with the $\text{SpanBERT}_{\text{Large}}$ encoder and (2) add a *turn* distance feature. The model for the Predicted phase and the Gold Mention phrase are both trained on automatically identified spans, while the model for the Gold Anaphor phase is trained on gold anaphors.
Markable extraction	For the Predicted phase, we employ the same mention extractor that we trained for the Anaphora Resolution track. For the Gold Mention and Gold Anaphor phases, gold mentions and gold anaphors are used as candidate anaphors respectively whereas gold mentions are used as candidate antecedents.
Training data	Three setups: (1) train on all of ARRAU 3.0; (2) pretrain on non-dialogue datasets (RST, GNOME, TRAINS91, TRAINS93), then train on data from the target (i.e., dialogue) domain ($\text{LIGHT}_{\text{train}}$, $\text{LIGHT}_{\text{dev}}$, $\text{AMI}_{\text{train}}$, AMI_{dev}, $\text{Persuasion}_{\text{train}}$, $\text{Persuasion}_{\text{dev}}$, $\text{Switchboard}_{\text{train}}$, $\text{Switchboard}_{\text{dev}}$); and (3) for each target dataset (e.g., LIGHT), first pretrain on non-dialogue datasets (RST, GNOME, TRAINS91, TRAINS93), then train on only the train split and the development split of the target dataset.
Development data	None: we perform parameter tuning directly on the test data.

Table 1: Overview of the approaches we adopted for the three tracks.

`coref-hoi` model. In order to help the reader understand the entity coreference model we employ for this year's shared task, we will begin by providing an overview of `coref-hoi` and UTD^{2021}.

2.1 `coref-hoi`

`coref-hoi` (Xu and Choi, 2020) is a re-implementation of the widely-used end-to-end coreference model by Lee et al. (2018). This model enumerates spans of up to a predefined length and, for computational efficiency reasons, generates a cultivated list of candidate mention spans that contains only a certain fraction n of the top spans, where n is a parameter known as the top span ratio. For each candidate mention span x, the model learns a distribution $P(y)$ over its candidate antecedents $y \in \mathcal{Y}(x)$. To maintain computational tractability, $\mathcal{Y}(x)$ contains only the top-k candidate antecedents (computed using the scoring function s_c, as described below) and a dummy antecedent ϵ, which should be selected when x does not have a coreferring mention preceding it in the associated text.

More specifically, $P(y)$ is computed as follows:

$$P(y) = \frac{e^{s(x,y)}}{\sum_{y' \in \mathcal{Y}(x)} e^{s(x,y')}}$$

where $s(x, y)$ is a pair-wise score that incorporates two types of scores: (1) $s_m(\cdot)$, a score that corresponds to the probability of a span being a mention, (2) $s_c(\cdot)$ and $s_a(\cdot)$, scores that correspond to the probability of two spans referring to the same entity ($s(x, \epsilon) = 0$ for dummy antecedents):

$$\begin{aligned} s(x, y) &= s_m(x) + s_m(y) + s_c(x, y) + s_a(x, y) \\ s_m(x) &= \texttt{FFNN}_m(g_x) \\ s_c(x, y) &= g_x^\top W_c g_y \\ s_a(x, y) &= \texttt{FFNN}_c(g_x, g_y, g_x \circ g_y, \phi(x, y)) \end{aligned}$$

where g_x and g_y denote the vector representations of x and y, W_c is a learned weight matrix for bilinear scoring, `FFNN(·)` denotes a feedforward neural network, $\phi(\cdot)$ encodes the distance between two spans as well as the meta-information such as speaker information.

While $s_c(\cdot)$ and $s_a(\cdot)$ both attempt to score a candidate antecedent given a candidate anaphor, $s_a(\cdot)$ is supposed to provide more accurate candidate antecedent scores. The reason is that $s_a(\cdot)$ is calculated using an `FFNN` while $s_c(\cdot)$ is a far less accurate bilinear scoring function. Nevertheless, $s_c(\cdot)$ is much more efficient to compute than $s_a(\cdot)$. Given its efficiency, $s_c(\cdot)$ is being used to score *all* candidate antecedents (i.e., all the spans preceding the candidate anaphor), and only the top-k scoring spans are used to compute $P(y)$ (where k is a tunable parameter). In other words, the computationally expensive-to-compute $s_a(\cdot)$ function only needs to be applied to the top-k candidate antecedents for each candidate anaphor.

2.2 UTD^{2021}

UTD^{2021} has the following four important extensions to `coref-hoi`:

Type prediction model Motivated in part by our previous work (Lu and Ng, 2020), we employ a type prediction model in UTD^{2021} that takes as input the span embedding g_x and computes the probability that span x has type t (i.e., $ot_x(t)$). The span type t_x is determined by the type with the highest probability. UTD^{2021} classifies each span into two types, NULL and ENTITY, where ENTITY covers both referring and non-referring mentions and NULL covers the spans that do not correspond to entities.

$$\begin{aligned} ot_x &= \texttt{FFNN}_t(g_x) \\ t_x &= \arg\max_t ot_x(t) \end{aligned}$$

A cross-entropy loss is computed using ot_x, which is then multiplied by a type loss coefficient and added to the loss function of `coref-hoi`. Specifically:

$$Loss = \lambda L_t + L_c$$

where L_t and L_c are the type prediction loss and the entity coreference loss respectively, and λ is the type loss coefficient, which specifies the relative importance of the two losses. In other words, UTD^{2021} jointly learns type prediction and entity coreference resolution. The motivation is to allow the two tasks to influence and mutually benefit from each other.

Sentence distance feature We hypothesize that recency plays a role in resolution, so we add the utterance distance between two spans as an extra feature into $\phi(x, y)$ in UTD^{2021}.

Span speaker constraint UTD^{2021} enforces a constraint on spans that is empirically derived from the training and development data: a span cannot cover more than one speaker's utterance.

Resolution constraint UTD^{2021} enforces a consistency constraint on resolution that will be used in both training and inference. This constraint uses simple heuristics designed for conversations to prevent two spans x and y from being posited as coreferent if they are *conflicting*. More specifically, we check whether a span belongs to one of the following eight groups:

1. span is or starts with: I, me, my, mine
2. span is or starts with: you, your, yours
3. span is or starts with: he, him, his
4. span is or starts with: she, her
5. span is: their
6. span is: it, its
7. span is: here
8. span is: there

Three constraints are applied to spans that belong to these groups:

C1 When two spans have the same speaker: if both of them are from groups 1, 2, 3, or 4 but they are not from the same group, then they cannot be coreferent.

C2 When two spans have different speakers: if both of them are from groups 1 or 2 and they are from the same group, then they cannot be coreferent.

C3 Regardless of the speakers: (1) *here* cannot be coreferent with *my*, *your*, *his*, *her*. and anything in group 5, group 6, and group 8; and (2) *there* cannot be coreferent with *my*, *your*, *his*, *her*, and anything in group 5, group 6, and group 7.

2.3 Our Approach

This year we develop a model for the Anaphora Resolution track that employs a three-step pipelined approach, which is composed of (1) mention extraction; (2) coreference resolution; and (3) removal of non-entity and non-referring mentions.

2.3.1 Step 1: Mention Extraction

The first step of our pipelined approach is to extract entity mentions from documents. As discussed before, `UTD`^{2021} performs joint entity coreference resolution and type prediction, where type prediction involves predicting each candidate mention span as ENTITY (referring/non-referring spans) or NULL (non-entity spans). We use `UTD`^{2021} for mention extraction as follows: all and only those candidate mention spans classified as ENTITY will be extracted as entity mentions and processed by the entity coreference model.

Recall that `UTD`^{2021} employs a loss function that is a weighted sum of the type prediction loss and the entity coreference loss, where the weight is determined by the type loss coefficient. To enable the model to focus on mention extraction (as opposed to entity coreference), we use with a large type loss coefficient. In addition, we disable the resolution constraints when applying `UTD`^{2021} in this step.

2.3.2 Step 2: Coreference Resolution

The second step of our pipelined approach is to produce coreference links using all and only those spans that are classified as ENTITY in the first step. To achieve this goal, we make the following changes to `UTD`^{2021} while keeping the span speaker constraint and resolution constraint.

Extracting candidate mention spans Instead of using span enumeration to generate candidate mention spans of up to a predefined length, we use the spans corresponding to gold entity mentions (including both referring or non-referring entity mentions) as the candidate mention spans during training and the spans corresponding to the mentions extracted in the first step as the candidate mention spans during testing.

Removing the type prediction model The type prediction model is no longer needed since the candidate mention spans are either gold spans (during training) or spans extracted in the first step (during testing). Hence, we simply remove it.

Removing the mention score Recall that in `coref-hoi`, the mention score $s_m(\cdot)$ indicates how likely a span corresponds to an entity mention. Since every candidate mention span is either a gold span (during training) or a span extracted in the first step (during testing), the mention score does not play a role anymore in determining how likely two candidate mentions are coreferent. Hence, we remove the mention score from the antecedent-anaphor pairwise score. So the new pairwise score $s(\cdot)$ becomes:

$$s(x, y) = s_c(x, y) + s_a(x, y)$$

where $s_c(\cdot)$ and $s_a(\cdot)$ are the same as those defined in `coref-hoi`.

Inference-time-only dummy antecedent rescoring Recall that in `coref-hoi`, the dummy antecedent is the correct antecedent for non-entity/non-anaphoric mentions. Based on empirical observations on our development data, our resolver fails to select the dummy antecedent as the antecedent for many non-anaphoric mentions. Consequently, we modify the score for dummy antecedents to make the model choose dummy antecedents more frequently. Specifically, instead of having $s(x, \epsilon) = 0$ for dummy antecedents, we make $s(x, \epsilon) = c$ $(c > 0)$ where c is a tunable parameter. By doing this, any candidate antecedent y of span x where $0 < s(x, y) < c$ will not be selected as an antecedent of x.

2.3.3 Step 3: Non-referring/Non-entity Mention Removal

The last step of our pipelined approach is to remove non-entity mentions and non-referring mentions. This step is motivated in part by our observation that our model achieves comparatively low CEAF$_e$ scores on the development data. We hypothesize that this was caused by the large number of erroneously identified singletons that correspond to non-referring or non-entity mentions. To address this problem, we train a model for identifying non-referring and non-entity mentions and apply it to the coreference output produced in Step 2 to remove singleton clusters containing these mentions. Specifically, we reuse our model in the first step, but instead of using span enumeration to generate candidate mention spans, we use gold entity mentions, gold non-referring mentions, and entity mentions in which the underlying word/phrase has appeared at least once as a gold entity mention in the training data as the candidate mention spans. The type prediction model is modified to predict two types: OTHER (for non-referring/non-entity spans) and REFERRING (for referring entity spans). Singletons that are predicted as OTHER are removed from the output.

2.4 Evaluation

In this subsection, we discuss some implementation details and the evaluation results of our system.

2.4.1 Corpora

We mainly use the given ARRAU 3.0 dataset (Uryupina et al., 2019), which contains two text corpora, RST and GNOME, and seven dialogue corpora, TRAINS91, TRAINS93, PEAR, LIGHT, AMI, Persuasion, and Switchboard. Each of the LIGHT, AMI, Persuasion, and Switchboard datasets contains a training set and a development set. Besides ARRAU, we use OntoNotes 5.0[1] to pretrain some of our models. We provide details about how we use these datasets in Section 2.4.2.

2.4.2 Implementation Details

We use SpanBERT$_{\text{Large}}$ as the encoder in all steps. We use different learning rates for the BERT-parameters and the task-parameters (1×10^{-5} and 3×10^{-4} respectively). In all three steps we train the model for 30 epochs with a dropout rate of 0.3. Each document in the training set is split into one or more training instances. Each training instance has at most five continuous segments, each of which contains 512 token pieces. We set n (the top span ratio) to 0.4 and k, the number of candidate antecedents for each candidate anaphor, to 50.

Prior to training on the shared task datasets, we pretrain both the first- and second-step models on OntoNotes 5.0. We do not pretrain our third-step model on OntoNotes because it covers only a portion of non-referring expressions. In fact, the only non-referring expressions covered by OntoNotes 5.0 are the predicate noun phrases, while we have a lot more in the shared task datasets (e.g., expletives, non-referring quantifiers, idioms).

[1]`https://catalog.ldc.upenn.edu/LDC2013T19`

2.4.3 Parameter Tuning

We divide the model parameters into two groups: those to be tuned on the development data and those to be tuned on the test data, as described below.[2]

Parameters tuned on the development data The set of parameters we tune on the development sets includes:

- the span width for span enumeration in the first step: we experiment with span widths out of {5, 10, 30};
- the number of epochs for pretraining the first- and second-step models: we search out of {10, 15, 20};
- the type loss coefficients (for the first- and third-step models): both type loss coefficients are searched out of {0.5, 1, 10, 100, 500, 800};
- the number of training epochs (for all models): we save a model checkpoint every five epochs and use the saved models to perform inference.

Parameters tuned on the test data In our final submissions, all development sets are also used as training data. The set of parameters we tune on the test set (using the model trained on both the training and development data) includes:

- the inference-time-only dummy antecedent re-scoring score (for the second-step model only): we experiment with integer scores between 0 and 10.
- the number of training epochs[3] (for all models): we save a model checkpoint every five epochs. Saved model checkpoints are used to do inference on test sets and inference output is evaluated by making a submission to the shared task competition.

Parameter tuning proceeds as follows. We tune the parameters associated with the three models in our pipeline in a *sequential* manner. Specifically, we first tune the parameters associated with the

[2]In principle, we are not supposed to tune parameters on the test data. We are effectively just exploiting the fact that we can evaluate our models on the test data by submitting our results to the submission site. While we could have tuned all the parameters on the test data, we did not do so because (1) it would take a lot of time to do so and (2) there is a limit on the number of submissions.

[3]Note that the number of training epochs is a parameter that appears in both groups: this parameter is first tuned on the development data and subsequently on the test data.

first-step model. Given the best parameter combination obtained for the first-step model, we then tune the parameters associated with the second-step model. Finally, given the best parameter combination obtained for the models in the first two steps, we tune the parameters associated with the third-step model.

Next, we describe how the parameters associated with each of the three models are tuned. For the first-step model, we first jointly tune the four development-set parameters. Then, using the max span width, the # of epochs for pretraining, and type loss coefficient obtained via this tuning process, we retrain the first-step model on the combined training and development data, tuning the number of training epochs on the test data.

Given the parameters tuned for the first-step model, we tune the parameters in the second-step model. As in the first-step model, we first jointly tune the four development-set parameters in the second-step model, then retrain the model using the best parameter combination on the combined training and development data, tuning the number of training epochs on the test data (assuming a dummy antecedent re-scoring score of 0). Finally, we tune the dummy antecedent re-scoring score.

Finally, given the parameters tuned for the models in the first two steps, we tune the parameters in the third-step model. The parameter tuning process for the third step model is the same as that for the first-step model.

2.4.4 System Variants

So far we have presented our coreference resolver as a three-step pipelined approach. In our evaluation, however, we test the following four variants of our approach:

1. `S1` corresponds to our model without the last two steps. In other words, we use only the first-step model to produce entity coreference results. Note that while the first-step model is intended for mention extraction, it performs joint type prediction and entity coreference resolution and therefore can be used to produce entity coreference results.
2. `S1,S2` corresponds to our model without the third step (removal of non-entity and non-referring mentions from the coreference output).
3. `S1,S3` corresponds to the setup where the coreference output produced by the first-step model is postprocessed by the third-step model to remove non-entity and non-referring mentions; and
4. `S1,S2,S3` is our full model.

	LIGHT	AMI	Pers.	Swbd.
`S1`	78.52	59.56	76.43	72.42
`S1,S2`	79.01	60.64	76.81	71.68
`S1,S3`	81.40	61.51	78.69	**75.81**
`S1,S2,S3`	**82.23**	**62.90**	**79.20**	75.25

Table 2: Anaphora resolution: evaluation results of the four variants of our approach expressed in terms of CoNLL score on the four test sets. The boldfaced results are our strongest results on the four test sets and hence our final results on the shared task competition leaderboard.

2.4.5 Results and Discussion

In this subsection, we report evaluation results obtained by making submissions to the shared task competition, which employs the Universal Anaphora Scorer[4] to calculate the CoNLL score, which is the unweighted average of the F-scores computed using the MUC, B^3, and $CEAF_e$ metrics.

Table 2 shows the entity coreference results on the official test data for the aforementioned four variants of our approach. A few points deserve mention. First, the `S1,S3` variant achieves the best result on Switchboard, while the `S1,S2,S3` variant achieves the best results on the remaining three test sets. Second, by comparing `S1` and `S1,S2`, we can see that `S2` yields only minor improvements (at most 1% CoNLL score) on three datasets and even adversely affects performance on Switchboard. We attribute the ineffectiveness of `S2` to the fact that `S1` has already produced good coreference links for the mentions it extracted. Thus, merely altering the coreference links would not bring much performance improvement. Third, by comparing `S1` and `S1,S3`, we can see that `S3` brings a 2%-3% CoNLL score improvement on all three datasets, which pinpoints one of the weaknesses of `S1` – having too many non-referring/non-mention spans in its prediction. The same conclusion can be drawn for `S2` by comparing `S1,S2` and `S1,S2,S3`.

Detailed evaluation results of the best performing system variant on each dataset in terms of MUC, B^3, and $CEAF_e$ precision (P), recall (R), and F-score (F) are shown in Table 3. As can be seen, the

[4]https://github.com/juntaoy/universal-anaphora-scorer

	MUC			B^3			$CEAF_e$			
	P	R	F	P	R	F	P	R	F	CoNLL
LIGHT	90.56	86.86	88.67	80.41	82.43	81.41	73.11	80.45	76.60	82.23
AMI	74.08	66.15	69.89	62.43	63.60	63.01	48.10	66.43	55.80	62.90
Persuasion	88.41	83.67	85.97	78.99	81.23	80.10	64.89	79.70	71.54	79.20
Switchboard	90.14	74.64	81.66	80.92	73.77	77.18	62.20	76.42	68.58	75.81

Table 3: Anaphora resolution: detailed evaluation results on the four test sets. These results are obtained using the system variant that achieves the best result on each test set.

(a) Official CoNLL scores of the system variants.

	S1				S1,S2				S1,S2,S3			
configuration	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
$\texttt{best}_{\texttt{dev}}$	77.95	58.99	75.71	71.79	78.63	59.18	76.33	71.46	82.18	62.72	79.02	74.86
$\texttt{best}_{\texttt{test}}$	78.52	59.56	76.43	72.42	78.80	60.64	76.72	71.68	-	-	-	-
$\texttt{best}_{\texttt{test}}$+DR	-	-	-	-	79.01	60.64	76.81	71.68	82.23	62.90	79.20	75.25

(b) Parameter settings for each system variant in different configurations.

		S1				S1,S2				S1,S2,S3			
configuration	parameter	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
$\texttt{best}_{\texttt{dev}}$	Maximum span width	30	30	30	30	-	-	-	-	-	-	-	-
	# of epochs for pretraining	15	15	15	15	15	15	15	15	-	-	-	-
	Type loss coefficient	500	500	500	500	-	-	-	-	500	500	500	500
	# of training epochs	15	15	15	15	10	20	10	5	5	5	10	10
$\texttt{best}_{\texttt{test}}$	Maximum span width	30	30	30	30	-	-	-	-	-	-	-	-
	# of epochs for pretraining	15	15	15	15	15	15	15	15	-	-	-	-
	Type loss coefficient	500	500	500	500	-	-	-	-	-	-	-	-
	# of training epochs	10	20	20	10	15	15	25	25	-	-	-	-
$\texttt{best}_{\texttt{test}}$+DR	Maximum span width	-	-	-	-	-	-	-	-	-	-	-	-
	# of epochs for pretraining	-	-	-	-	15	15	15	15	-	-	-	-
	Type loss coefficient	-	-	-	-	-	-	-	-	500	500	500	500
	# of training epochs	-	-	-	-	15	15	25	25	20	30	5	25
	Dummy antecedent re-scoring	-	-	-	-	3	0	1	0	-	-	-	-

Table 4: Anaphora resolution: official CoNLL scores and detailed parameter settings of three system variants.

performance on AMI is much worse than the performance on any other datasets. We speculate that the poor performance on AMI is related to its comparatively longer documents, as long dependencies are hard for the model to learn.

To better understand the impact of parameter tuning on the resolution performance of the system variants, we report in Tables 4a and 4b the official CoNLL scores and parameter settings for three configurations:

- $\texttt{best}_{\texttt{dev}}$ corresponds to the configuration that yields the highest CoNLL score on the test data when only the development-set parameters are tuned;
- $\texttt{best}_{\texttt{test}}$ corresponds to the configuration that yields the highest CoNLL score on the test data when the development-set parameters and one of the test-set parameters (the number of training epochs) are tuned; and
- $\texttt{best}_{\texttt{test}}$+DR corresponds to the configuration that yields the hgihest CoNLL score on the test data when the development-set parameters and both of the test-set parameters are tuned. Note that $\texttt{best}_{\texttt{test}}$+DR and $\texttt{best}_{\texttt{test}}$ differ only in terms of whether the dummy antecedent re-scoring constant is tuned after all the remaining parameters are tuned.

Table 4b reports the parameters as follows. First, the parameters reported for S1, S1,S2, and S1,S2,S3 are the parameters obtained for the first-step model, the second-step model, and the third-step model, respectively. Since the parameter associated with these three models are tuned in a sequential fashion, the full set of parameters for S1,S2,S3 can be recovered from the table by combining parameters from S1, S1,S2, and S1,S2,S3. Second, the first three rows for the three configurations are the same, as those parameters are tuned on the development data only. Third, $\texttt{best}_{\texttt{test}}$+DR for S1 is not applicable, as dummy antecedent re-scoring is used in the second-step model. Moreover, we do not report the results of S1,S3 in this table. Because of time limitations we do not perform parameter tuning for this variant: for the parameters associated with S1 we simply reuse the parameters we tuned for the S1 variant, and for the parameters associated with S3 we set the type loss coefficient to 500 and the number of training epochs to 10.

Several observations can be made on the results in Table 4.[5] First, $\texttt{best}_{\texttt{test}}$ outperforms $\texttt{best}_{\texttt{dev}}$ consistently for a 0.2-1% in CoNLL score, showing that parameter tuning on the test data does lead to performance improvements. Second, dummy antecedent re-scoring is not very effective in improving resolution performance. Comparing $\texttt{best}_{\texttt{test}}$ and $\texttt{best}_{\texttt{test}}$+DR for our S1,S2 model, we see that dummy antecedent re-scoring brings only a diminutive CoNLL score improvement of 0.1-0.2% on two test sets and no improvement at all on the remaining two.

We conclude this section by mentioning that while our systems ranked first among all participants in the anaphora resolution track, there are still some weaknesses in our systems. First, our systems have a hard time handling long dependencies, which we hypothesize to be the main reason why our systems performed the worst on AMI. Second, our system cannot handle cases of plural anaphoric reference in which the antecedents are introduced by separate mentions, namely split antecedents.

3 Discourse Deixis Resolution

The Discourse Deixis track in this year's shared task has three evaluation phases: (1) the Predicted phase, where a system needs to extract both antecedents and anaphors and perform discourse deixis resolution; (2) the Gold Mention phase, which is the same as the Predicted phase except that anaphors are to be extracted from the given set of gold mentions; and (3) the Gold Anaphor phase, which is the same as the Gold Mention phase except that gold anaphors are explicitly given. The Gold Anaphor phase is introduced in this year's shared task to partially address the difficulty of comparing different resolvers with respect to their *resolution* performance (Li et al., 2021).

[5]In Table 4a the $\texttt{best}_{\texttt{test}}$ results for S1,S2,S3 are not available due to time limitations.

3.1 Approach

We cast discourse deixis resolution as identity anaphora resolution. This allows us to use Xu and Choi's (2020) coref-hoi model as our baseline for discourse deixis resolution. In this section, we describe our approach to discourse deixis resolution, which is composed of six extensions to coref-hoi.

1. Candidate Anaphor Extraction In the shared task datasets, most deictic expressions are demonstrative pronouns (e.g., "that", "this") and "it". These three pronouns account for more than 80% of the anaphors in the given datasets. Thus, we impose a simple heuristic to extract candidate anaphors: instead of extracting them by span enumeration, we only allow a span in which the underlying word/phrase has appeared at least once in the training set to be a candidate anaphor.

2. Anaphor Prediction Similar to our discourse deixis resolution system in the CODI-CRAC 2021 shared task (Kobayashi et al., 2021), we use a type prediction model in our system this year. Different from last year, however, the type prediction model is used to identify those candidate anaphors that correspond to deictic expressions. Thus, only two types are used: ANAPHOR (the candidate anaphor is indeed a deictic expression) and NULL (the candidate anaphor is not).

3. Candidate Antecedent Extraction Since the shared task datasets are annotated in a way so that only utterances can serve as an antecedent of deictic expressions, we extract candidate antecedents as follows. For each span i that is predicted as ANAPHOR by the type prediction model, we select the 10 utterances that are closest to i (including the utterance in which i appears) as its candidate antecedents. The motivations are that (1) deictic expressions are anaphoric expressions, and hence recency plays an important role in antecedent selection, and (2) using the 10 closest utterances allows us to cover more than 95% of the antecedent-anaphor pairs in the datasets.

4. Dummy Antecedent Elimination In coref-hoi, the set of candidate antecedents for every span includes a dummy antecedent, which

Type	Features
Anaphor	Embedding of the sentence the anaphor is in
Antecedent	# of words; # of nouns; # of verbs; # of adjectives; # of content word overlaps between antecedent and the preceding words of the anaphor; whether an antecedent is the longest among all candidate antecedents; whether an antecedent has the most content word overlap among all candidate antecedents
Pairwise	Sentence distance between a candidate antecedent and an anaphor, ignoring sentences that contain only interjections, filling words, reporting verbs, and punctuation

Table 5: Additional features used in our model.

will be selected as the antecedent of a span i if (1) i is not an entity mention or (2) i is an entity mention but it is not anaphoric.

For our model, the situation is different. Since only those spans predicted as ANAPHOR by the anaphor prediction model will be passed to the antecedent selection model, the antecedent selection model only sees spans that have been classified as anaphoric. Since these spans are anaphoric, they should presumably not be resolved to the dummy antecedent. For this reason, we eliminate the dummy antecedent from the set of candidate antecedents of every span when training and testing the antecedent selection model.

5. Features Our next extension involves a large-scale expansion of features, hypothesizing that hand-engineered features could be profitably used by a span-based model. Specifically, we incorporate three types of features: (1) anaphor-based features, which encode the context of an anaphor, (2) antecedent-based features, which encode some statistics computed based on a candidate antecedent, and (3) pairwise features, which encode the relationship between an anaphor and a candidate antecedent. The list of features is shown in Table 5. We add these features to both the bilinear score $s_c(x, y)$ and the concatenation-based score $s_a(x, y)$:

$$s_c(x, y) = g_x^\top W_c g_y + g_s^\top W_s g_y$$
$$s_a(x, y) = \texttt{FFNN}_c(g_x, g_y, g_x \circ g_y, g_s, \phi(x, y))$$

where W_c and W_s are learned weight matrices, g_s is the embedding of the sentence s in which anaphor x appears, and $\phi(x, y)$ encodes the speaker information as well as different types of distance between x and y.

6. Inference-Time-Only Distance-Based Candidate Antecedent Filtering Given that we have fewer training instances for those antecedent-anaphor pairs that have larger sentence distances and it is generally harder to learn long-distance dependencies, correctly resolving an anaphor whose antecedent is far away from it is by no means easy. Although we use only the 10 closest utterances during training, we propose to further lower this number during inference. Specifically, for each candidate anaphor, the model selects an antecedent from one of the n closest utterances ($1 \leq n < 10$), where n is a tunable parameter.

3.2 Evaluation

In this subsection, we evaluate our system and discuss the implementation details.

3.2.1 Implementation Details

The models we use in the three evaluation phases are similar. Specifically, the only difference between our models in different phases lies in Extension 1 (candidate anaphor extraction). In the Predicted phase, candidate anaphors are selected using the method stated in Extension 1. In the Gold Mention phase, the candidate anaphors used for both training and inference are those words/phrases in the given set of gold mentions that appeared in the training set as deictic expressions. In the Gold Anaphor phase, we use the given anaphors for both training and inference, so there is no need to extract anaphors.

We use SpanBERT$_{\text{Large}}$ as the encoder for all evaluation phases. We use different learning rates for the BERT-parameters and the task-parameters (1×10^{-5} and 3×10^{-4} respectively). Each document in the training set is split into one or more training instances. Each training instance has at most 12 continuous segments, each of which contains 512 tokens. Models are trained for 30 epochs with a dropout rate of 0.3.

Note that the models used for the later phases were retrained given the gold mentions and gold anaphors.

3.2.2 Parameter Tuning

Given that we can make submissions to the shared task competition and the amount of data we have is far from abundant, we use all the given datasets as our training set, and tune the following three parameters on the test data (by submitting the system output to the shared task competition):

	MUC			B^3			$CEAF_e$			
	P	R	F	P	R	F	P	R	F	CoNLL
Predicted Phase										
Light	37.04	31.25	33.90	50.80	33.43	40.32	60.77	26.65	37.05	37.09
AMI	51.67	52.54	52.10	58.76	51.75	55.04	65.06	44.41	52.79	53.31
Persuasion	48.44	59.05	53.22	56.38	57.10	56.74	62.34	47.34	53.82	54.59
Switchboard	63.77	41.12	50.00	70.62	39.28	50.48	76.52	35.82	48.79	49.76
Gold Mention Phase										
Light	37.17	32.81	34.85	51.59	35.53	42.08	59.81	28.07	38.21	38.38
AMI	54.46	51.69	53.04	63.15	51.87	56.96	69.31	46.07	55.35	55.12
Persuasion	50.00	58.10	53.74	58.16	56.15	57.13	64.52	46.14	53.80	54.89
Switchboard	66.67	42.99	52.27	71.08	39.35	50.65	72.29	34.35	46.57	49.83
Gold Anaphor Phase										
Light	46.88	46.88	46.88	65.13	50.56	56.93	77.02	40.88	53.41	52.40
AMI	71.19	71.19	71.19	81.05	69.12	74.61	87.47	60.76	71.71	72.50
Persuasion	67.62	67.62	67.62	80.42	67.30	73.28	87.10	55.68	67.93	69.61
Switchboard	70.09	70.09	70.09	80.03	69.83	74.58	86.36	61.25	71.67	72.11

Table 6: Discourse deixis resolution: official results on the test sets.

- the type loss coefficient: we search out of {0.5, 1, 5, 10, 100, 500, 800} using grid search.
- the inference-time-only candidate antecedent filtering constant: we experiment with all integers between 1 and 10.
- the number of training epochs: we save a model checkpoint every five epochs and evaluate it on the test set.

We jointly tune the type loss coefficient and the number of training epochs, and determine the candidate antecedent filtering constant after the other two parameters are fixed.

3.2.3 Results and Discussion

We report the detailed official evaluation results of our system for different phases in Table 6. A few points deserve mention. First, by comparing the results in the Predicted phase and the Gold Mention phase, we can see that even though the set of candidate anaphors is being narrowed down in the Gold Mention phase, only a small performance gain (at most 1% CoNLL score) is achieved. We speculate that our simple heuristic for selecting candidate anaphors is effective, so the provision of gold mentions does not eliminate many plausible candidate anaphors. Second, the provision of gold anaphors has brought huge improvements (14%-22% CoNLL score) to our system, which shows that one of the key weaknesses of our system is anaphor identification. Third, across all three phases, our system performs much worse on LIGHT than on other datasets. Further investigations are needed to determine the reason.

To better understand the impact of parameter tuning on the test data, we show in Tables 7a and 7b the CoNLL scores achieved by three system configurations on the test data:

- $\texttt{worst}_{\texttt{test}}$ corresponds to the configuration that yields the worst result on the test data when only the number of training epochs and the type loss coefficient are jointly tuned (i.e., the antecedent filtering constant is simply set to 10);
- $\texttt{best}_{\texttt{test}}$ corresponds to the configuration that yields the best result on the test data when only the number of training epochs and the type loss coefficient are jointly tuned (i.e., the antecedent filtering constant is simply set to 10);
- $\texttt{best}_{\texttt{test}}\texttt{+AF}$ corresponds to the configuration that yields the best result on the test data when the inference-time-only antecedent filtering constant is tuned, with the other two parameters taken from $\texttt{best}_{\texttt{test}}$.

Several observations can be made on the results shown in Table 7. First, $\texttt{best}_{\texttt{test}}$ outperforms $\texttt{worst}_{\texttt{test}}$ consistently for at most 8% in terms of CoNLL score. The biggest performance gap of 7.97% is observed on the Switchboard test set in the Gold Mention phase: as can be seen, the parameters associated with the two configurations differ only with respect to the number of training epochs. This suggests that the number of epochs plays an important role in the performance of our discourse deixis resolver. Similar conclusions can be drawn by comparing the results in other phases and on other test sets. Second, inference-time-only antecedent filtering generally offers little perfor-

(a) Official CoNLL scores of our models.

	Predicted Phase				Gold Mention Phase				Gold Anaphor Phase			
configuration	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
$\texttt{worst}_{\texttt{test}}$	34.58	47.36	48.78	45.67	34.07	47.68	49.47	41.86	51.77	68.65	66.78	69.26
$\texttt{best}_{\texttt{test}}$	37.09	51.48	50.30	47.96	37.89	55.12	53.40	49.83	52.40	72.50	69.61	72.11
$\texttt{best}_{\texttt{test}}$+AF	37.09	51.61	50.42	47.96	38.38	55.12	54.89	49.83	52.40	72.50	69.61	72.11

(b) Parameter settings for each setup in different phases.

		Predicted Phase				Gold Mention Phase				Gold Anaphor Phase			
configuration	parameter	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
$\texttt{worst}_{\texttt{test}}$	Type loss coefficient	0.5	0.5	0.5	0.5	500	100	0.5	500	800	800	800	800
	# of training epochs	10	10	15	10	5	15	15	5	10	15	20	15
	Antecedent filtering constant	10	10	10	10	10	10	10	10	10	10	10	10
$\texttt{best}_{\texttt{test}}$	Type loss coefficient	0.5	0.5	0.5	0.5	800	500	500	500	800	800	800	800
	# of training epochs	10	5	20	20	15	10	10	15	15	5	15	10
	Antecedent filtering constant	10	10	10	10	10	10	10	10	10	10	10	10
$\texttt{best}_{\texttt{test}}$+AF	Type loss coefficient	0.5	0.5	0.5	0.5	800	500	500	500	800	800	800	800
	# of training epochs	10	5	20	20	15	10	10	15	15	5	15	10
	Antecedent filtering constant	10	7	7	10	7	10	7	10	10	10	10	10

Table 7: Discourse deixis resolution: official CoNLL scores of our models and detailed parameter settings in different phases.

mance improvement, though it has yielded performance gains of 0.2%-1.3% in CoNLL score on some test sets.

We conclude this section by pointing out that our system ranked first for all three phases in the discourse deixis resolution track. While our system was 1%-5% CoNLL scores better than the second-ranked team in the Predicted phase and the Gold mention phase, our system outperformed the second-ranked team by large margins of 5%-16% CoNLL scores in the Gold Anaphor phase, which shows the effectiveness of our system in discourse deixis resolution.

4 Bridging Resolution

Like the Discourse Deixis track, the Bridging Resolution track in this year's shared task has three different phases, namely the Predicted phase, the Gold Mention phase, and the Gold Anaphor phase. While discourse deixis resolution has received fairly little attention in the NLP community in recent years, constant progress has been made for bridging resolution. Nevertheless, such progress has thus far limited to a large extent to the Gold Mention setting, where gold mentions are given, and the Gold Anaphor setting, where gold anaphors are given (see Kobayashi and Ng (2020) for a comprehensive overview and Kobayashi et al. (2022a) for state-of-the-art results). In particular, little progress has been made on end-to-end bridging resolution, which corresponds to the setup used in the Predicted phase of the shared task.

Motivated in part by the success of the hybrid rule-based and learning-based approach to bridging resolution developed by Kobayashi and Ng (2021), we adopted a multi-pass sieve approach to bridging resolution in last year's shared task, where we employed a pipeline of sieves consisting of a neural sieve, which is essentially Yu and Poesio's span-based neural model that employs multi-task learning, and a set of same-head sieves, which were specifically designed to target the identification of bridging links between two mentions having the same head. Given that the improvement offered by the same-head sieves is small, we abandon them this year and focus instead on extending Yu and Poesio's multi-task learning framework for bridging resolution. Below we first provide an overview of Yu and Poesio's model.

4.1 Yu and Poesio's (2020) Model

Yu and Poesio's (Y&P) model is a span-based neural model that takes gold mentions as input and jointly performs entity coreference resolution and bridging resolution. The way Y&P differs from other end-to-end span-based coreference models is that it uses two `FFNN`'s to separately predict coreference links and bridging links. These two `FFNN`s share the first few hidden layers as well as the span representation layer. The loss function of this MTL model is composed of a weighted sum of the losses of the bridging task and the coreference task. Unlike feature-based approaches to bridging resolution, where feature engineering plays a critical role in performance, this neural model employs only two features, the length of a mention and the mention-pair distance.

4.2 Approach

Since Y&P's model takes gold mentions as input, we need a mention extractor before we can deploy it. For this reason, we employ a pipelined approach to bridging resolution, where we first extract mentions using a mention extractor and then perform bridging resolution using our extended Y&P model. Below we describe the extensions we made to Y&P.

4.2.1 Extensions to the Y&P Model

We employ two extensions to the Y&P model.

Using SpanBERT as encoder Given the successful application of SpanBERT to entity coreference in the past few years, it is natural to think about applying SpanBERT to bridging resolution. In fact, SpanBERT has recently been shown to yield promising results when applied to resolving bridging references in narratives (Kobayashi et al., 2022b). Hence, our first extension to Y&P involves replacing its biLSTM encoder and the frozen BERT/Glove embeddings used by the biLSTM with SpanBERT$_{\text{Large}}$ in order to strengthen Y&P's performance. We adopt the independent version of Joshi et al. (2019), where each input document is split into non-overlapping segments of length up to L_s.

Adding Turn Distance as a feature As mentioned above, Y&P employs only two features, namely the length of a mention and the mention-pair distance. Since Y&P is not designed for the dialogue domain, neither of the two features captures information regarding the dialogue domain. We follow our work in last year's shared task and add the turn distance between mentions as a feature, where a turn is defined as a set of contiguous sentences by the same speaker.

4.3 Evaluation

In this subsection, we evaluate our system and discuss the implementation details.

4.3.1 Implementation Details

Each document is split into segments of length 384. The 40% top scoring spans are retained for bridging resolution. The weight parameter associated with the weighted sum of losses of the bridging task and the coreference task is set to 1, meaning that the two tasks are given equal importance in the learning process. Below we discuss how the models used for the three phases differ from each other.

4.3.1.1 Predicted Phase

In the Predicted phase, our system needs to extract mentions and perform bridging resolution. We first use our `S1` system described in Section 2 to extract mentions, then use our modified Y&P model to perform bridging resolution on the extracted mentions.

We test our model with the following training setups as different setups may lead to large performance differences:

`T1`: In this setup, we use all the available datasets for model training, namely ARRAU RST, GNOME, TRAINS91, TRAINS93, LIGHT, AMI, Persuasion, and Switchboard. In particular, both the training split and the development split of LIGHT, AMI, Persuasion, and Switchboard are used for training. Our system is trained for at most 25 epochs.

`T2`: In this setup, we first pretrain our system on the datasets that are outside of the target (i.e., dialogue) domain, namely ARRAU RST, GNOME, TRAINS91, and TRAINS93, for 15 epochs. After that, we train our system on one dataset that contains all of the data from the target domain, namely LIGHT, AMI, Persuasion, and Switchboard, for 25 epochs.

`T3`: Similar to `T2`, we first pretrain our system on the datasets that are outside of the target domain for 15 epochs. However, for each dataset from the target domain, we train our model for 25 epochs using both the training split and the development split of that target domain. For instance, when evaluating our system on LIGHT$_{\text{test}}$, we train a model on LIGHT$_{\text{train}}$ and LIGHT$_{\text{dev}}$. Hence, the documents used to train the models in `T3` are a subset of those used to train the models in `T2`.

In preliminary experiments, we found that models trained with both predicted mentions and gold mentions performed better than models trained with only gold mentions. Thus, for each training setup, we first extract mentions from the training

		Light			AMI			Persuasion			Switchboard		
		P	R	F	P	R	F	P	R	F	P	R	F
Predicted Phase													
Recognition	T1	56.80	26.23	35.89	42.46	17.59	24.88	39.35	35.86	37.52	55.30	25.86	35.24
	T2	53.37	33.13	40.88	37.44	18.29	24.57	41.46	39.14	40.27	46.48	28.45	35.29
	T3	46.20	41.13	43.52	43.87	15.74	23.17	35.09	43.75	38.95	48.93	34.48	40.46
Resolution	T1	34.93	16.13	22.07	22.91	9.49	**13.42**	27.80	25.33	26.51	27.65	12.93	17.62
	T2	30.36	18.84	**23.25**	18.48	9.03	12.13	28.57	26.97	**27.75**	22.89	14.01	17.38
	T3	24.07	21.43	22.67	22.58	8.10	11.93	24.80	30.92	27.53	23.85	16.81	**19.72**
Gold Mention Phase													
Recognition	T1	61.66	23.77	34.31	52.76	24.31	33.28	44.36	40.13	42.14	53.99	18.97	28.07
	T2	57.85	35.84	44.26	43.55	25.00	31.76	46.02	43.75	44.86	49.16	31.47	38.37
	T3	56.04	34.85	42.98	40.38	34.49	37.20	41.27	49.01	44.81	49.44	37.72	42.79
Resolution	T1	39.30	15.15	21.87	31.16	14.35	**19.65**	32.36	29.28	30.74	31.29	10.99	16.27
	T2	34.99	21.67	**26.77**	21.77	12.50	15.88	34.60	32.89	33.73	26.26	16.81	20.50
	T3	33.86	21.06	25.97	18.70	15.97	17.23	31.86	37.83	**34.59**	26.27	20.04	**22.74**
Gold Anaphor Phase													
Recognition	T1	97.78	97.78	97.78	97.69	97.69	97.69	98.03	98.03	98.03	98.49	98.49	98.49
	T2	97.78	97.78	97.78	97.69	97.69	97.69	98.03	98.03	98.03	98.49	98.49	98.49
	T4	97.78	97.78	97.78	97.69	97.69	97.69	98.03	98.03	98.03	98.49	98.49	98.49
Resolution	T1	46.80	46.80	**46.80**	39.35	39.35	**39.35**	56.58	56.58	56.58	43.75	43.75	43.75
	T2	40.15	40.15	40.15	31.71	31.71	31.71	51.97	51.97	51.97	37.07	37.07	37.07
	T4	46.55	46.55	46.55	38.19	38.19	38.19	56.91	56.91	**56.91**	44.40	44.40	**44.40**

Table 8: Bridging resolution: recognition results and resolution results on the test sets. The boldfaced results are the official F-scores of our system on the shared task leaderboard.

set using our S1 system[6] and then use the extracted mentions along with the gold mentions for model training.

4.3.1.2 Gold Mention Phase

In the Gold Mention phase, we do not retrain our models. Instead, we perform bridging resolution on the given gold mentions in the test data using the models trained in the Predicted phase.

4.3.1.3 Gold Anaphor Phase

In the Gold Anaphor phase, since gold anaphors are explicitly given, we constrain our models so that only gold anaphors can be resolved to other gold mentions during both training and inference. We test our models using the T1 and T2 setups mentioned in Section 4.3.1 as well as a new setup:

T4: After training our model in the T1 setup, we execute an extra fine-tuning step where we fine-tune our model for 25 epochs using both the training split and the development split of the target domain. For instance, when evaluating our system on LIGHT$_{\text{test}}$, we first train a model using the T1 setup and then fine-tune the resulting model on LIGHT$_{\text{train}}$.

T4 serves as an alternative to T3. The only difference between T3 and T4 is that T3 performs fine-tuning on the target domain *after* it finishes pretraining on datasets outside of the target domain, whereas T4 performs extra fine-tuning on the target domain after a model is trained according to T1.

4.3.2 Parameter Tuning

We do not tune any parameters on the development data. The number of training epochs is the only parameter we tune on the test data. As in anaphora resolution and discourse deixis resolution, to tune the number of training epochs we save a model checkpoint every five epochs and evaluate it on the test set. Note that the number of training epochs is tuned separately for each setup.

4.3.3 Results and Discussion

For each test set, the best resolution result achieved over all setups will be used as our official result. Table 8 shows the official recognition and resolution results of our bridging resolver on the test sets. Our system achieves resolution F-scores of 13.42%-27.75% for the Predicted phase. For the Gold Mention phase and Gold Anaphor phase, our

[6]Note that S1, which was trained on the training set, is applied to the training set to extract mentions.

(a) Predicted phase

	T1, Predicted Phase				T2, Predicted Phase				T3, Predicted Phase			
# epochs	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
5	17.44	13.42	24.78	13.46	23.20	10.73	24.78	19.23	22.67	11.93	25.09	17.68
10	17.75	11.98	25.27	15.76	22.24	10.49	26.05	17.12	22.58	11.17	27.53	19.72
15	22.07	13.27	26.51	17.62	23.25	12.13	26.73	17.38	21.48	11.73	26.64	16.16
20	20.61	11.73	24.59	17.54	22.06	11.91	27.75	17.08	22.66	11.24	26.94	16.24
25	21.43	12.48	22.80	17.33	23.12	11.73	27.42	17.26	21.52	10.88	25.65	17.85

(b) Gold Anaphor phase

	T1, Gold Anaphor Phase				T2, Gold Anaphor Phase				T4, Gold Anaphor Phase			
# epochs	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.	LIGHT	AMI	Pers.	Swbd.
5	46.80	37.73	56.58	43.75	38.92	28.47	48.36	32.76	46.55	37.04	53.62	44.40
10	46.31	39.35	48.68	42.67	40.15	31.71	51.32	37.07	46.55	36.11	53.62	42.24
15	45.94	37.04	51.32	42.24	38.55	30.56	50.00	36.21	46.55	37.27	54.93	42.03
20	45.07	37.04	52.63	43.32	36.95	30.79	51.97	35.99	46.55	38.19	56.91	43.10
25	44.83	38.43	52.30	42.46	37.68	29.86	50.33	35.34	-	-	-	-

Table 9: Bridging resolution: official resolution F-scores of our models in terms of the number of training epochs and the setup for two phases.

system achieves F-scores of 19.65%-34.59% and 39.35%-56.91% respectively. The performance improvements in the later phases should not be surprising, as the task becomes progressively easier in the later phases.

A few points deserve mention. First, the results show a strong positive correlation between recognition performance and resolution performance. This should not be surprising either, as strong recognition performance lays the foundation for strong resolution performance. Second, as mentioned above, we do not retrain our models in the Gold Mention phase. Thus, the performance gains we achieve in the Gold Mention phase over the Predicted phase can be attributed solely to the difference between using predicted mentions and using gold mentions. In particular, almost all setups achieve better performance in the Gold Mention phase except T1 on Switchboard, where worse results are obtained for both recognition and resolution performance. We speculate that, although gold mentions are given, identifying bridging anaphors is still a non-trivial task. Additional experiments are needed to determine the reason, however. Third, in the Gold Anaphor phase, all setups achieve much better results than those in the Gold Mention phase. In some setups the results increase by 100%. This should not be surprising, as anaphor recognition performance has gone from around 30% F-score to nearly 100% F-score.

To examine the impact of parameter tuning on the test data, we show in Tables 9a and 9b how the resolution F-score of our bridging resolver on the test data varies with the number of training epochs for each setup. Note that these results are available only for the Predicted phase and the Gold Anaphor phase but not the Gold Mention phase because in the Gold Mention phase we simply reuse the models trained during the Predicted phase. As we can see, the number of training epochs has a large impact on the performance of our bridging resolver: the difference in resolution F-score between the worst combination and the best combination can be as large as 4.63%. The choice of setup can lead to a even larger difference — an F-score difference of 11.64% between T2 and T4 on Switchboard$_{test}$ in the Gold Anaphor phase.

We conclude this section by mentioning that our system ranked first in all phases of the bridging resolution track. In particular, our system outperformed the second-ranking team for 4%-9% resolution F-scores in the Gold Anaphor phase.

5 Conclusions

We presented the systems that we developed for all three tracks of the CODI-CRAC 2022 shared task, namely the anaphora resolution track, the bridging resolution track, and the discourse deixis resolution track. For anaphora resolution, we employed a three-step approach consisting of mention extraction, coreference resolution, and removal of non-referring and non-entity mentions. Our results demonstrated that the third-step model, the non-referring/non-entity removal model, contributed a lot to overall resolution performance. However, our system is still not able to handle split-antecedents, which is a direction for future improvements. For discourse deixis resolution, our results revealed that one of the key weaknesses in our system is anaphor detection, as a large performance gain could be achieved when the model was applied to gold anaphors. For bridging resolution, our results showed that the Gold Anaphor phase was much easier than the Predicted phase and the Gold Mention phase. The resulting large performance gap provided suggestive evidence that there is still a lot of room for improvement in bridging anaphor detection. Future work should focus on (1) determining the extent to which performance would deteriorate when all model parameters are tuned on development data and (2) performing a cross-team analysis to better understand how the resolvers from different teams are different from each other.

Acknowledgments

This work was supported in part by NSF Grant IIS-1528037. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the NSF.

References

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

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.

Sopan Khosla, Juntao Yu, Ramesh Manuvinakurike, Vincent Ng, Massimo Poesio, Michael Strube, and Carolyn Rosé. 2021. The CODI-CRAC 2021 shared task on anaphora, bridging, and discourse deixis in dialogue. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 1–15, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hideo Kobayashi, Yufang Hou, and Vincent Ng. 2022a. Constrained multi-task learning for bridging resolution. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 759–770, Dublin, Ireland. Association for Computational Linguistics.

Hideo Kobayashi, Yufang Hou, and Vincent Ng. 2022b. End-to-end neural bridging resolution. In *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Hideo Kobayashi, Shengjie Li, and Vincent Ng. 2021. Neural anaphora resolution in dialogue. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 16–31, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hideo Kobayashi and Vincent Ng. 2020. Bridging resolution: A survey of the state of the art. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3708–3721, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hideo Kobayashi and Vincent Ng. 2021. Bridging resolution: Making sense of the state of the art. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1652–1659, Online. 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.

Shengjie Li, Hideo Kobayashi, and Vincent Ng. 2021. The CODI-CRAC 2021 shared task on anaphora, bridging, and discourse deixis resolution in dialogue: A cross-team analysis. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 71–95, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2020. Conundrums in entity coreference resolution: Making sense of the state of the art. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6620–6631, Online. Association for Computational Linguistics.

Olga Uryupina, Ron Artstein, Antonella Bristot, Federica Cavicchio, Francesca Delogu, Kepa Joseba Rodríguez, and Massimo Poesio. 2019. Annotating a broad range of anaphoric phenomena, in a variety of genres: the ARRAU corpus. *Natural Language Engineering*, 26:95 – 128.

Liyan Xu and Jinho D. Choi. 2020. Revealing the myth of higher-order inference in coreference resolution. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8527–8533, Online. Association for Computational Linguistics.

Liyan Xu and Jinho D. Choi. 2021. Adapted end-to-end coreference resolution system for anaphoric identities in dialogues. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 55–62, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Neural mention detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1–10, Marseille, France. European Language Resources Association.

Juntao Yu, Sopan Khosla, Ramesh Manuvinakurike, Lori Levin, Vincent Ng, Massimo Poesio, Michael Strube, and Carolyn Rosé. 2022. The CODI-CRAC 2022 shared task on anaphora, bridging, and discourse deixis in dialogue. In *Proceedings of the CODI-CRAC 2022 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Juntao Yu and Massimo Poesio. 2020. Multitask learning-based neural bridging reference resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3534–3546, Barcelona, Spain (Online). International Committee on Computational Linguistics.