Anaphora Resolution in Dialogue: Description of the DFKI-TalkingRobots System for the CODI-CRAC 2021 Shared-Task

Tatiana Anikina, Cennet Oguz, Natalia Skachkova
Siyu Tao, Sharmila Upadhyaya, Ivana Kruijff-Korbayová
DFKI / Saarland Informatics Campus, Saarbrücken, Germany
ivana.kruijff@dfki.de

Abstract

We describe the system developed by the DFKI-TalkingRobots Team for the CODI-CRAC 2021 Shared-Task on anaphora resolution in dialogue. Our system consists of three subsystems: (1) the Workspace Coreference System (WCS) incrementally clusters mentions using semantic similarity based on embeddings combined with lexical feature heuristics; (2) the Mention-to-Mention (M2M) coreference resolution system pairs same entity mentions; (3) the Discourse Deixis Resolution (DDR) system employs a Siamese Network to detect discourse anaphor-antecedent pairs. WCS achieved F1-score of 55.6% averaged across the evaluation test sets, M2M 57.2% and DDR 21.5%, all on predicted mentions.

1 Introduction

This paper describes the anaphora resolution system developed by the DFKI-TalkingRobots Team for the *CODI-CRAC 2021 Shared-Task* (CCST)[1] (Khosla et al., 2021). Anaphora resolution (AR) is important for many natural language processing tasks, such as information extraction, summarization, question answering, etc. Most existing systems have been developed for AR in text and focus on coreference resolution. CCST was set up to address AR in dialogue, including not only coreference but also bridging and discourse deixis.

We were inspired to join CCST, although we did not have an existing AR system to start with. Initial small tests of publicly available pre-trained AR models from the AllenNLP and HuggingFace libraries on our dialogue data in the domain of disaster response team communication (Skachkova and Kruijff-Korbayová, 2020) yielded discouraging results. Participating in CCST was a way to kick start the development of an AR system of our own. Our long-term aim is to develop a domain-independent AR system to be used in several projects of the Talking Robots Group at DFKI[2], including for the interpretation of team communication in disaster response (Willms et al., 2019).

Track	AR	AR	DD
System	WCS Sec. 2.1	M2M Sec. 2.2	DDR Sec. 2.4
Setting	pred.	pred.	pred.
Baselines	–	–	–
Learning framework	workspace clustering	mention to mention pairing	heuristics Siamese Net
Markable identification	SpaCy	BiLSTM-CRF	SpaCy
Train. data	CCST	CCST	CCST
Dev. data	CCST	CCST	CCST

Table 1: Systems summary

Our system for the CCST is the first step in this direction, and should be considered work in progress. It consists of two independent AR subsystems: *Workspace Coreference Resolution System* (WCS, Sec. 2.1); *Mention-to-Mention Coreference Resolution System* (M2M, Sec. 2.2); and the *Discourse Deixis Resolution System* (DDR, Sec. 2.4).The subsystems were developed using the data provided for CCST and evaluated in the CCST tracks *Eval-AR* and *Eval-DD (Pred)*, respectively. See Tab. 1 for a summary.

Sec. 2 describes the subsystems one by one, including the respective approach, mention detection method and how we used the CCST data. We also highlight the differences between our approaches and recent related work in the respective subsections. In Sec. 3 we provide the results and error analysis for each subsystem. We conclude and point out future work in Sec. 4.

2 System description

2.1 Workspace Coreference System (WCS)

We designed and implemented a system for identity anaphora resolution based on the following

[1]URL: competitions.codalab.org/competitions/30312

[2]URL: dfki.de/mlt

Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue, pages 32–42
Punta Cana, Dominican Republic, November 10, 2021.

principles. First, our system processes the input incrementally to make the learning process resemble the way humans read and understand text and resolve coreference. Second, we restrict the number of possible coreference clusters considered at each step to only the salient candidates (i.e., those clusters that have been referred to recently in the conversation). Inspired by the clustering ideas outlined in Rahman and Ng (2011) our system creates clusters on-the-fly by comparing the current mention to the most recent members from each of the clusters available in the workspace.

We combined workspace-based clustering with a binary classification of referring expressions. Fig. 1 shows an overview of the whole system that considers one mention at each step and computes the probabilities for clustering this mention with other mentions available in the workspace.

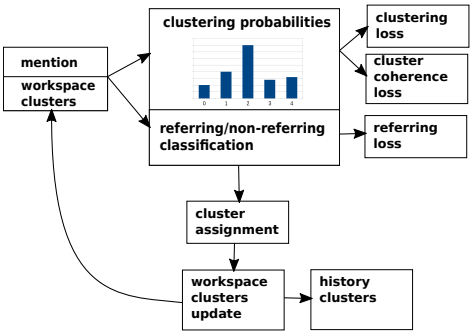

Figure 1: Workspace Coreference System Overview

Our implementation differs from other submissions to the Shared Task. Both winning models (lxucs and UTD_NLP) are based on the pairwise scoring of mentions and extend the coreference model from (Xu and Choi, 2020). The model submitted by KU_NLP is based on pointer networks whereas our system is more similar in spirit to the incremental approach introduced in (Xia et al., 2020). Similarly to their work we are building clusters of mentions incrementally and remove old mentions from the workspace if the corresponding clusters have not been updated for a long time.[3]

However, there are some important differences between our model and the work by Xia et al. (2020). For example, we do not use separate representations for clusters, but rather take the latest 5 mentions from each cluster and compute an average probability of assigning a new mention to the same cluster. Moreover, Xia et al. (2020) evict clusters based on their size and distance while our decision to remove a cluster from the workspace is based on the number of singletons that are allowed in the workspace as well as recency of the cluster updates. We also have substantial differences in the training procedure, e.g., rather than considering all correct antecedents, Xia et al. (2020) focus only on the most recent one while our system checks all antecedents, i.e., compares how many mentions in the assigned cluster are actually coreferent with the mention. We also update the parameters after each clustering decision and not once per document. Moreover, our model does not use SpanBERT embeddings or any other components pre-trained on coreference data[4] and is more focused on exploring different kinds of embeddings and how their combinations contribute to the coreference resolution.

2.1.1 Data

We used the data provided by the CCST organizers and split it into train and development set for WCS as follows. First, we extracted individual documents from each of the files (RST_DTreeBank, Gnome_Subset2, Pear_Stories, Trains_91, Trains_93, AMI_dev, LDC2021E05_Switchboard_3_dev, light_dev and Persuasion_dev). Then we randomly shuffled these documents to create a training set with 550 documents and a development set with 61 documents. We did not use any additional data for training/development. The system was tested on the CCST test set in the Eval-AR track.

2.1.2 SpaCy-based Mention Detection

We use SpaCy[5] to extract mention candidates (markables) for WCS. The candidates include all noun chunks detected by SpaCy (based on the *en_core_web_trf* model). We add all pronouns and adverbs like *'here'*, *'there'*, *'now'* and *'then'* because they are not recognized by SpaCy as noun phrases. We also check for each noun phrase whether it has a prepositional phrase or clause (based on the dependency relations *'prep'* and *'relcl'* in SpaCy) and extend the span if necessary.

After the release of the CCST annotated test set we checked the performance of our SpaCy-based approach by comparing the gold spans to those extracted by our system. We got the following

[3]The limit was set to 100 steps without updates in our experiments.

[4]Xia et al. (2020) used encoder weights that were already fine-tuned on the OntoNotes dataset and adopted both mention and pairwise scorers from (Joshi et al., 2020).

[5]https://spacy.io

micro-F1 scores: 72.4% on AMI, 85.6% on Light, 81.6% on Persuasion and 81.5% on Switchboard. These results show that the quality of the mention extraction significantly impacts the overall performance of the coreference resolution system. E.g., WCS had the lowest F1 score on AMI (43.93%) and the highest score on Light (67.28%), which is in line with the mention extraction performance.

2.1.3 WCS Input and Architecture

After extracting mentions we pre-process all sentences in the documents using BERT (pre-trained cased version) and extract various features for each mention, including the mention position within the document, the head lemma, head number,[6] animacy classification,[7] speaker embedding, BERT embedding (Devlin et al., 2018) of the mention's head, averaged BERT embedding for the span of the mention, Numberbatch concept embedding (Speer et al., 2017) for the head lemma, GloVe embedding (Pennington et al., 2014) for the head lemma, averaged GloVe span embedding and the masked language model (MLM) embedding for the mention. For the MLM embedding we use BERT (pre-trained cased base model) and take the current and the previous dialogue turns, mask the mention's head in the turn and ask the model to predict the top candidate for the masked token. We take the GloVe embedding of this candidate as the MLM embedding.

When processing a new mention we compare its representation to each of the workspace clusters. Note that the very first position in the workspace is always reserved for the singleton case. All embeddings for the singleton case have zero values but keep the same dimensions as the real cluster members. In order to compare a mention to the workspace clusters we took up to 5 latest members from each cluster and computed the probability for each member-mention pair. The individual pair-based probabilities were averaged to obtain the cluster-level probabilities as shown in Fig. 2.

When training the model we used the extracted features of different types and concatenated each mention feature with the corresponding member feature (e.g., mention speaker with the member speaker embedding) before passing them through the linear layers and activation functions to obtain the final embedding. A separate list of layers (a.k.a. channels) was used for each embedding type. We

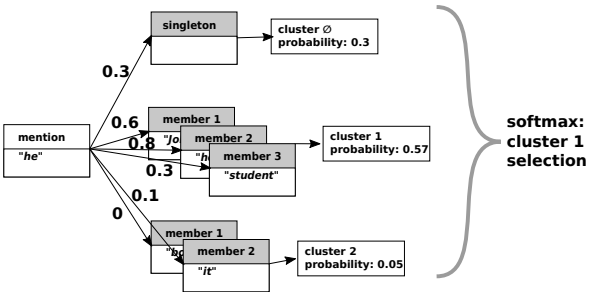

Figure 2: Workspace Clustering Example

concatenated the output of different channels and passed it to the feed-forward network. We used ReLU as an activation function for most of the channels. ReLU was also used for the final decision (when combining the outputs of different channels). In parallel, we performed a binary referring expression classification using a network with two linear layers. This network helped the model to decide whether an input mention is referring or not. If it was classified as non-referring it was not clustered with any of the workspace clusters.

After obtaining a probability for each of the workspace clusters as shown in Fig. 2 we assign the mention to the most likely cluster. Then we add the new mention to the workspace clusters, either as a new member of some existing cluster or as a new singleton cluster. In each iteration the clusters in the workspace have to be re-arranged. If some clusters have not been updated for more than 100 steps they are removed from the workspace and stored in the history. Additionally, we limit the maximum number of singleton clusters in the workspace to 20. When deciding whether to put the cluster in the history or keep it in the workspace we sort all candidates by the recency of the last update.

We experimented with different kinds of embeddings and their combinations on a subset of the data. The details are presented in A.

2.1.4 WCS Training

In order to provide a learning signal for the network we combined three different loss functions. The first loss compares the gold clusters to the ones generated by the model (clustering loss). The second loss checks how heterogeneous are the mentions that appear in the same cluster (cluster coherence loss) and the third loss is a simple cross-entropy loss for the binary classification of referring/non-referring mentions (referring loss).

While training our model we used a teacher forcing ratio of 0.3 to enforce the correct clustering in

[6] We use SpaCy Lemmatization and number annotation.

[7] We trained a separate classifier for this using the CCST entity type annotations.

30% of all cases and make the network converge faster. We also checked the compatibility between the clusters based on the features typically used in coreference systems. These features include number and animacy compatibility and nesting. E.g., number compatibility guarantees that mentions in plural are not clustered together with the ones that are in singular and animacy compatibility prevents clustering animate and inanimate objects together. Nesting simply means that if some mention is within the NP represented by another mention these two should not be clustered together (e.g., *'a good student with a book'* and *'a book'*). These conditions were added in order to make the learning process easier by filtering out incompatible clusters at an early stage. However, since our animacy classification and number annotations were not perfect, the filtering stage could cause some mistakes in the later clustering. Additionally, we checked first person pronouns and speaker consistency. E.g., if the current cluster has a speaker A with the mention *'I'*, another mention *'I'* with a different speaker B cannot be put in the same cluster.

2.1.5 WCS Post-processing

At test time we used an additional constraint that all referential mentions that are third person pronouns such as *'it'* and *'they'* cannot form a new singleton cluster and must be clustered with one of the preceding workspace clusters. In order to score the workspace clusters for this task we ordered them by recency starting with the most recent ones and checked the mention-cluster compatibility as explained above (using animacy, number and nesting constraints). Among the compatible clusters we selected the one that had the highest probability according to the masked language model. For this we masked the current mention in the original sentence and checked one by one the likelihoods of mentions from different clusters to fill the position. These likelihoods were averaged for each cluster based on the 5 most recent members and finally the cluster with the highest likelihood was selected. We trained WCS for 5 epochs with a learning rate of 0.0001 and *Adam* as an optimizer.

2.2 Mention-to-Mention System (M2M)

Coreference resolution systems aim to find the mentions of the same entities which might be referred to by different expressions (e.g., synonyms, pronouns, etc.) or the same expressions according to their presented context in the document. This is challenging because changes in the context and mentions of the entities strongly affect the coreference resolution systems. Additionally, the same expressions might also refer to different entities in the real world. These noted points cause three different challenges that the M2M model aims to overcome. The first is pairing the same mentions of different entities because of the same context. The second is the problem with the different contexts of the same entities when they are referred to by the same expressions. The third one is the different mentions of the same entities in different contexts.

Most coreference resolution models have been adapted to rank the antecedents based on average embeddings (Lee et al., 2017, 2018) and contextualized embeddings (Joshi et al., 2019, 2020). Furthermore, Xu and Choi (2020) has adapted the model implementation from (Lee et al., 2018) and used the contextualized span embeddings (Joshi et al., 2020) to obtain the mention candidates through span enumeration and aggressive pruning. The M2M system combines the average and contextualized embeddings instead of using them alone, to avoid context variation of the same mentions and capture lexical similarities of different mentions. Therefore, the M2M system is designed with the aim of suppressing the context variations of different or the same entities. This simply means that the different mentions might be paired because their contexts are similar, and on the contrary, the same expressions might not be paired together because their context or tokens are different. Therefore, we aim first to investigate the effect of mention-mention matching, and then pooling them together with the paired entities into the same clusters. Therefore, the M2M system is designed to cluster the mentions which are paired with the common candidates.

Whereas the M2M implementation differs from the system of pointer network by KU_NLP, it is similar to the winning systems lxucs and UTD_NLP because it is designed to match the most similar entities with a pairwise comparison. M2M, however, forms clusters based on transitive closure over the mention-mention pairs, i.e., pairs which have a common mention are clustered together. Additionally, our mention representation method differs by the included distance vector.

2.2.1 Data

To train the M2M model we use the Light, Persuasion, and Switchboard data provided by the CCST organizers. All three datasets are used to train the

Pronoun and *Noun Models* (see below). For training the *Self Model* (see below) the Switchboard data is not used, because the speaker of the corresponding utterances is not given in the training data, only in test data. Therefore, only the Light and Persuasion data is used to train the *Self Model*.

2.2.2 Mention Detection with BiLSTM-CRF

We treat mention detection as a sequence tagging task, analogous to the common approach to Named Entity Recognition. Given an input sequence of tokens, our mention detection model extracts markable chunks by tagging the tokens in the IOB2 format, i.e., with the beginning of each chunk tagged B, any subsequent tokens in the chunk tagged I, and tokens not in any chunk tagged O.

We implemented a Bidirectional LSTM - Conditional Random Field (BiLSTM-CRF) Model, shown to be effective in sequence tagging (Huang et al., 2015). As the standard IOB2 format only annotates flat chunks, our model only predicts the markables with the widest scope in cases of nesting. We flattened the training data accordingly when converting it to the IOB2 format.

The architecture of the model is as follows: the input data (in IOB2 format) is first passed through an ELMo layer (Peters et al., 2018), where contextualized word representations (embeddings) were extracted for each word. The ELMo embeddings are then passed to the BiLSTM-CRF model. Finally, the CRF model predicts the tags for the input tokens. In a post-processing step, we used SpaCy (*en_core_web_trf*) to extract any nested noun chunks from the markables predicted by the model. Maximum sentence length was set to 50, and we used the *Adam* optimizer with a learning rate of 0.0005 and trained the model for 4 epochs.

The training and development sets are created as an 8:2 split of the documents in each CCST dataset. This was done to include sentences from all datasets and prevent bias towards one dataset. The label distribution is shown in Tab. 2.

Tag	# in data	F1 score
O	139,703	0.99
B	55,947	0.90
I	136,752	0.92

Table 2: BiLSTM-CRF : label distribution and F1-score on CCST train/development data

While our model achieves an overall F1 score of 0.98 on the development set, a breakdown of the score by label (Tab. 2) shows that the F1 score for B and I tags are significantly lower, at 0.92 and 0.90, respectively. This means that our current model is overly conservative in predicting the markable chunks, indicated by a large number of false negatives for the B and I tags. The false negatives result in incomplete markables being extracted and impact the performance of the downstream task.

2.3 M2M Training

We use contextualized word embeddings BERT and ELMo, and average word embeddings GloVe and Fasttext (Joulin et al., 2016). The individual utterances are considered as the input for the embedding systems to extract the vectors for the head and span of the mentions. In order to obtain the rich feature representation of the mentions, word embedding models are used under four different contributions: (1) the 10-*th* layer of BERT is used to obtain a significant local representation without global context of utterances; (2) ELMo is employed for positional encoding of full context of the utterances; (3) GloVe is applied to represent the external global context of each token; (4) Fasttext is utilized to extract the n-gram features of words. Because a mention can be represented with different language forms such as a span of multiple words, one word entity, pronoun, etc., we used different embeddings to obtain the vectors: only BERT is used for span representation, while a combination of BERT, ELMo, GloVe, and Fasttext is used for head representation of each mention.

Each mention is represented by using the word embedding methods with four different feature sets: (1) head embedding to represent similarities between the different mentions of the same entities; (2) span embeddings to represent the differences between similar mentions of the different entities; (3) the distance feature vector is obtained by using distance functions between the head embeddings of mentions to represent the similarities via a distance 7-*d* vector; (4) speaker embeddings by using the average embeddings.

Three different models are designed as part of the M2M system. The *Self Model* is built for personal pronouns with average embeddings of the speaker and head features of the mentions. The *Pronoun Model* is designed to pair the third person and place pronouns to the corresponding candidate by exploiting the head average embeddings of mentions. The *Noun Model* attempts to find the pairs of

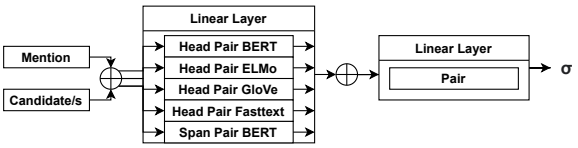

Figure 3: M2M Noun Model Architecture and Training

the mentions by using head and span embeddings with contextualized and average embeddings when they are noun phrases. Fig. 3 shows the Noun Model.

2.3.1 M2M Architecture and Training

The models are designed with linear layers with a ReLU activation function for the head, speaker and span features. Then the extracted vectors from embeddings are concatenated with a distance vector to train a linear layer with a ReLU activation function. Then a linear layer with a Sigmoid activation function is used for output probabilities.

The models are trained with the same hyperparameters. The learning rate is defined as 0.0001 and the optimizer is *Adam*. For computing the loss, we specify *BCELoss* which is a criterion that measures the Binary Cross Entropy between the target and the output. Training is done for 30 epochs.

2.4 Discourse Deixis Resolution(DDR)

Webber (1991) defines discourse deixis as reference to a discourse entity such as a proposition, description, event, speech act, etc. The discourse entity is a part of the discourse model of the dialogue participant and is realized in a certain discourse segment, i.e. a chunk of a linguistic text, e.g., a sequence of clauses or utterances. The exact boundaries of discourse segments are often hard to define even for humans (Webber, 1991; Artstein and Poesio, 2006). In addition, the resolution of discourse deixis mentions requires cross-sentence information about rhetorical relations and focus/attention which is often not annotated. All this makes discourse deixis resolution a very challenging task.

There exist only a few frameworks focusing on the resolution of discourse deixis mentions. Most early works that we know of (e.g., Eckert and Strube (2000), Navarretta (2000), or Byron (2002)) are rule-based. They use hand-crafted rules to pick out anaphors, which are usually demonstrative pronouns. The right antecedent for each anaphor is found relying on candidates' dialogue act types. Early machine learning approaches are presented, e.g., by Strube and Müller (2003) and Müller (2008), who use many elaborate manually created feature vectors to represent markables, and train classifiers to predict whether an antecedent-anaphor pair is valid or not. Marasović et al. (2017) introduce a model that relies on a neural network (a Siamese Net) to rank potential antecedent candidates given an anaphor. The input for the network, i.e. sentences containing anaphors and antecedent candidates, is represented as feature vectors (embeddings) using bi-LSTM.

Works focusing on automatic identification of discourse deixis anaphors are also not many. An algorithm based on hand-crafted features to recognize discourse deictic *it*, *this* and *that* is suggested, e.g., by Müller (2008). Other works, e.g., by Eckert and Strube (2000), Kolhatkar et al. (2018), or Roussel et al. (2018) present analyses of anaphors with non-nominal antecedents and their properties, but no actual implementations.

Our approach combines machine learning and hand-crafted rules. The machine learning part is inspired by the mention-ranking model for abstract anaphora resolution by Marasović et al. (2017). Given a discourse deixis mention that needs to be resolved, the task is to choose the most suitable antecedent among several candidates. To do so, we train a scorer implemented as a Siamese Net with the goal to assign a higher score to true anaphor-antecedent pairs and lower scores to false ones. Our implementation of the Siamese Net is much simpler than that of Marasović et al. (2017) - instead of bi-LSTM we use BERT's [CLS] token to extract utterance embeddings, and our antecedent-anaphor score is defined as simple Euclidean distance between the two vectors produced by the Siamese Net. Identification of anaphors and the antecedent candidates is rule-based. We explain the details of our system below.

2.4.1 Data

The scorer is trained on all the CCST train and development data. Before that the light_dev file was used to work on the heuristic rules. We built our dataset as follows. For each utterance containing a discourse deixis anaphor (anchor utterance) we create a pair of examples (see Fig. 4) by replacing the anaphor with its true antecedent (a positive example) and with a false candidate (a negative example), which is usually the utterance before the true antecedent. The idea is to make the Siamese Net learn that the true antecedent fits semantically into the utterance containing the anaphor. In total

Anchor: *That* certainly scared the eternal daylights out of me !
Positive example: *I myself happened to look out from the kitchen and see his dangling legs in front of the window* . certainly scared the eternal daylights out of me !
Negative example: *He never experimented with flying contraptions again* . certainly scared the eternal daylights out of me !

Figure 4: Discourse deixis dataset instance

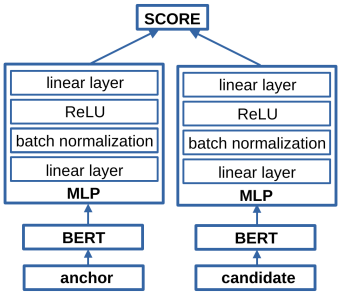

Figure 5: DDR scorer architecture

the dataset contains 5,171 examples.

2.4.2 Discourse Deixis Mention Detection

We do not assume that markables are given during testing, so it is our task to recognize and cluster them. For this we employ heuristic rules that are based on observations in the light_dev data.

First, we want to find discourse deixis mentions that need to be resolved. For now we focus only on demonstrative pronouns, which we detect using the SpaCy NLP pipeline (Honnibal et al., 2020). SpaCy does not distinguish between different types of non-personal pronouns, and labels them all with a DET tag. So, we consider all subtrees with a DET head linked to a parent node via *'dobj'*, *'pobj'*, *'nsubj'* or *'attr'* relations as discourse deixis markables that need to be resolved.

Next, we need to find the antecedents of the detected discourse deixis mentions and cluster the right mentions together. It should be noticed that, in most cases these clusters only include 2-3 mentions, and the antecedents can be found within the local context of several previous utterances. We assume that the antecedents are verbal phrases, clauses, or sequences of clauses (split antecedent cases).

For extracting the antecedent candidates we consider the current utterance containing the mention, and 3 previous ones. Each previous utterance, as well as their continuous sequences are viewed as possible (split) antecedents. Besides, we find all subtrees with VERB or AUX heads linked to the ROOT node via *'relcl'*, *'conj'*, *'ccomp'*, or *'advcl'* relations. We remove all overlapping candidates and candidates that occur after the mention to be resolved. In case of nested candidates, the longest one is chosen. We also apply the following simple heuristics to exclude unlikely candidates and improve the results. First, we limit the size of the clusters by two elements. Here we also consider cases of anaphors referring to split antecedents, which form their own singleton clusters. Second, we disregard all candidates that do not have verbal roots.

2.4.3 DDR Architecture and Training

The discourse deixis resolution system ranks the antecedent candidates using a scorer, sorts the scores in descending order and selects the top candidate. At this step we limit the output of the scorer to 5 candidates and re-sort them giving preference to the candidates closest to the anaphor, as usually the anaphor immediately follows its antecedent.

Being a Siamese Net, the scorer consists of two independent pipelines that have identical architectures. One pipeline is designed for the anchor, the other for an antecedent candidate. Each pipeline accepts input in the form of BERT input IDs, which are then fed into the $\text{BERT}_{\text{BASE}}$ model. We use the models' pooler output to represent each sequence, which is next processed by a simple multilayer perceptron (MLP) network containing two linear layers with a batch normalization and a ReLU activation function. Finally, the Euclidean distance between the feature representations produced by both MLP networks is calculated, and the pair with the smallest distance gets the biggest score. The scorer's architecture is shown in Fig. 5.

The scorer is trained for 12 epochs. We use adaptive learning rate starting with 3e-5, the *AdamW* optimizer and the *TripletMarginLoss* loss function.

3 Results and Error Analysis

3.1 Workspace Coreference System (WCS)

WCS achieved an average F1-score of 64.99% on Light, 43.93% on AMI, 59.93% on Persuasion and 53.55% on Switchboard. Although the results are lower than those of the top-performing systems, we believe that our approach presents a new way to treat coreference as an incremental clustering task

and can be further improved by experimenting with different configurations of input and hyperparameters as well as more robust feature extraction.

We believe that our system should benefit from better animacy and number annotations because these are crucial for the decision on the mention-cluster compatibility as explained in Section 2.1. Using a better mention extraction model should further improve the scores. We also evaluated WCS using the gold mentions as input and obtained the following results: 73.24% on Light, 56.48% on AMI, 72.06% on Persuasion and 62.85% on Switchboard. The difference in scores shows that WCS gains up to 12% F-score on AMI and Persuasion, up to 9% on Switchboard and 6% on Light when it uses the correct mentions. We noticed that quite often markables extracted by SpaCy were incomplete and sometimes even missing. E.g., NPs such as *'my two year old'* and *'the whole time'* were not recognized as NPs by SpaCy and some parts of complex NPs such as *'the kind of things you're supposed to look for'* were missing. Nested NPs present another big challenge for SpaCy, e.g., *'case'* in *'the case material'* was not recognized as a markable. Our feature extraction process heavily relies on SpaCy because we use it together with some heuristics to find a head in each mention span and perform the lemmatization. For instance, in one case *'someones'* was selected as a head in *'someones free will'*, hence the incorrect embedding for *'someone'* instead of *'will'* was used for the clustering decision.

Some of the rules and heuristics we implemented for WCS do not apply in some cases. E.g., the AMI dataset proved to be especially challenging because the dialogues in AMI have several participants and there are references to objects that were not previously mentioned in the text. For instance, a third person pronoun *'it'* in *'do you wanna hold it'* does not have an antecedent in the preceding text because the object can be inferred from the physical context. But our algorithm requires to find an antecedent for each mention of *'it'* that is a referring expression, hence *'it'* ends up in the wrong cluster.

We plan to continue our experiments with WCS by improving the feature extraction process and applying WCS to different domains and genres.

3.2 Mention-to-Mention System (M2M)

The M2M system achieved F1-score of 61.26% on Light, 59.20% on Persuasion and 51.24% on Switchboard. Whereas *Self Model* shows high pairing performance, we observe that *Pronoun Model* struggles with lexical variation between pronouns and the corresponding nouns. We assume that the high performance of *Self Model* is based on the speaker invariant, i.e. two speakers occur in the data. For example, the model pairs the mentions *'I'* and *'my'* of a speaker always with the corresponding speaker. On the other hand, the delicate performance with the *Pronoun Model* is observed based on the high lexical variation: *'He'* needs to be paired with the *'father'* as well as *'his'*. However, the design of M2M is not able to handle more than one possible pair even though the correct pair count is more. The *Noun Model* is also affected by this problem.

To conclude, M2M performs effectively with the similar lexical features of mention-candidate pairs where the candidate variation is lower, e.g., personal pronouns of speakers. However, it struggles to pair the mentions when they are represented with different lexical features. At the same time, It suffers from the same mentions of different entities. In order to overcome the deficiencies of the M2M system, we expect that the clustering mechanism solves the lexical variation problem between mentions by obtaining one possible cluster instead of many possible candidate mentions.

3.3 Discourse Deixic Resolution (DDR)

The DDR system achieved F1-score 20.97% on the Light dataset, 17.43% on AMI, 23.76% on Persuation and 23.86% on Switchboard in the Eval-DD (Pred) track. The submitted system combines the scorer and four heuristic rules partially described in Sec. 2.4, namely limiting cluster size to two mentions, removing all non-verbal antecedent candidates, restricting the scorer's output to five candidates and re-scoring them by distance to the anaphor. For the error analysis we use fully annotated test data released after the competition ended. We perform discourse deixis mention detection as presented in Sec. 2.4.2, and check the performance of the system adding rules incrementally. These results are shown in Tab. 3. The first row of shows the F1 scores achieved by the scorer on the three fully annotated test sets. The next two rows present the scores for cases when the scorer is not forced to search for clusters larger than two mentions, and is applied after the removal of non-verbal candidates. The last four rows show the results after re-sorting the scorer's output by distance to the anaphor. N

Corpus	light	AMI	Persuasion
bare scorer	15.65	13.27	23.36
+ limiting cluster size	16.21	12.99	23.82
+ removing non-verbal	17.11	16.99	**26.48**
+ sort by dist. ($N = 3$)	21.38	17.36	26.06
+ sort by dist. ($N = 4$)	21.32	**17.62**	24.31
+ sort by dist. ($N = 5$)	**21.85**	17.16	23.60
+ sort by dist. ($N = 6$)	20.79	17.22	23.63

Table 3: DDR F1 scores on fully annotated test sets

stands for the number of candidates ranked by the scorer that we re-sort.

Without additional heuristic rules the performance of the scorer is really poor. We assume that the main reasons for this are as follows. First, our model does not possess any information about the semantics of separate tokens and utterances. The example in Fig. 4 illustrates the problem. Using only local context, it is impossible for the model to understand why *'seeing his dangling legs in front of the window'* is a better candidate for being scary than *'never experimenting with flying contraptions'*. BERT, which we use to get utterance representations, is not capable of providing such deep semantics. In the result, both positive and negative examples look legitimate for the model.

Second, the model also lacks information about the discourse structure and how different utterances relate to each other, and thus has difficulties recognizing split antecedents.

Next, our simple approach has a problem with both recall and precision. Recall is bad because mentions with determiner heads make up only 13%-35% (depending on the sub-corpus) of all discourse deixis mentions in the data. Precision is negatively affected by the fact that the DET tag in SpaCy is very general. As a result, we extract many false discourse deixis mentions, like relative pronoun *'that'* in *"I use it to clean the tower that I work in ."* or a determiner phrase *'some of these beautiful flowers'* which is a case of 'standard' anaphora.

Using heuristic rules mostly has a positive effect up to 5% on the test sets. However, not all of these rules work well with different dialogue data. Some sub-corpora contain a lot of fragments, many of which are rather short. As a result, the true antecedent may lie more than three utterances back. Some utterances are ungrammatical, and SpaCy makes mistakes parsing them, which leads to wrong candidate boundaries. Searching for antecedent candidates, we try to split complex and compound utterances. This often leads to rather cumbersome SpaCy-based rules that easily fail, because dialogue speech patterns are very diverse, so it is impossible to create an exhaustive set of rules to capture them. Our focus on verbal (split) antecedents may also lead to errors - we work with transcribed dialogue data, and antecedents are not necessarily verbal, they can also be represented by noun phrases like *'the inscriptions on the side of the fountain'*, prepositional phrases *'to Bath'*, adverbial phrases *'u : and back to Avon'*, etc. Finally, re-sorting the scorer's output by distance does not work for all types of dialogue data (see Tab. 3).

We plan to continue working on discourse deixis resolution, as our naive system offers some space for improvement. Recall and precision of discourse deixis mentions are first obvious aspects to work on. For this purpose current approaches to shell nouns resolution and event detection can be considered.

4 Conclusions and Outlook

We described the three anaphora resolution subsystems we developed from scratch for CCST using only the datasets provided by the CCST organizers. WCS is a coreference resolution approach that incrementally clusters mentions using semantic similarity based on embeddings combined with lexical feature heuristics. It employs SpaCy for markable detection. M2M is another coreference resolution approach which pairs the mentions for the same entities. For markable detection it uses our own BiLSTM-CRF model trained on the CCST data. WCS achieved F1-score of 55.6% averaged across the CCST evaluation test sets, using predicted markables; the M2M system achieved 57.2%. Finally, DDR employs a Siamese Network to distinguish true discourse anaphor-antecedent pairs. Its markable detection is also based on SpaCy. It has average F1-score 21.5% using predicted mentions.

Our models employ novel ideas, but are still in the early stages of development. CCST gave us a boost and we see many options for future work regarding both markable detection as well as the AR resolution models themselves. For example, we would like to improve the feature extraction for WCS; enhance M2M by clustering; combine WCS and M2M; and improve the detection of discourse deixis mentions that need to be resolved. We also plan to apply the systems on other datasets, which we are currently preparing.

References

Ron Artstein and Massimo Poesio. 2006. *Identifying reference to abstract objects in dialogue*. Universität Potsdam.

Donna Karen Byron. 2002. *Resolving pronominal reference to abstract entities*. University of Rochester.

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

Miriam Eckert and Michael Strube. 2000. Dialogue acts, synchronizing units, and anaphora resolution. *Journal of Semantics*, 17(1):51–89.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. SpaCy: Industrial-strength Natural Language Processing in Python.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging.

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.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. FastText.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

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*. Association for Computational Linguistics.

Varada Kolhatkar, Adam Roussel, Stefanie Dipper, and Heike Zinsmeister. 2018. Anaphora with non-nominal antecedents in computational linguistics: A survey. *Computational Linguistics*, 44(3):547–612.

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. ACL.

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. ACL.

Ana Marasović, Leo Born, Juri Opitz, and Anette Frank. 2017. A mention-ranking model for abstract anaphora resolution. *arXiv preprint arXiv:1706.02256*.

Mark-Christoph Müller. 2008. *Fully automatic resolution of'it','this', and'that'in unrestricted multi-party dialog*. Ph.D. thesis, Universität Tübingen.

Costanza Navarretta. 2000. Abstract anaphora resolution in danish. In *1st SIGdial Workshop on Discourse and Dialogue*, pages 56–65.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Altaf Rahman and Vincent Ng. 2011. Narrowing the modeling gap: A cluster-ranking approach to coreference resolution. *J. Artif. Intell. Res.*, 40:469–521.

Adam Roussel, Stefanie Dipper, Sarah Jablotschkin, and Heike Zinsmeister. 2018. Towards the automatic resolution of anaphora with non-nominal antecedents: Insights from annotation.

Natalia Skachkova and Ivana Kruijff-Korbayová. 2020. Reference in team communication for robot-assisted disaster response: An initial analysis. In *Proceedings of the 3rd Workshop on Computational Models of Reference, Anaphora and Coreference. Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC-2020), befindet sich COLING'2020, December 12, Virtual, Spain*. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4444–4451. AAAI Press.

Michael Strube and Christoph Müller. 2003. A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 168–175.

Bonnie Lynn Webber. 1991. Structure and ostension in the interpretation of discourse deixis. *Language and Cognitive processes*, 6(2):107–135.

Christian Willms, Constantin Houy, Jana-Rebecca Rehse, Peter Fettke, and Ivana Kruijff-Korbayová. 2019. Team communication processing and process analytics for supporting robot-assisted emergency response. In *IEEE SSRR*.

Patrick Xia, João Sedoc, and Benjamin Van Durme. 2020. Incremental neural coreference resolution in constant memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8617–8624.

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. ACL.

Acknowledgements

The authors are supported by the German Ministry of Education and Research (BMBF): T. Anikina and I. Kruijff-Korbayová in project CORA4NLP (grant Nr. 01IW20010); C. Oguz in IMPRESS (grant Nr. 01IS20076) and N. Skachkova in A-DRZ (grant No. I3N14856).

A Appendix: WCS Embeddings

We experimented with using different kinds of embeddings and their combinations for the WCS subsystem on a subset of the data. We used RST_DTreeBank_dev, Switchboard_3_dev and Persuasion_dev files for training and evaluated performance on the light_dev data. In total, we used 50 documents for training and 20 for testing. Unlike other participants (lxucs and UTD_NLP) we did not use SpanBERT embeddings, but we explored GloVe, BERT, Numberbatch, masked LM embeddings as well as separate embeddings for speaker, position and distance between mentions.

Embeddings:	1	2	3	4	5	6	7	8	9	10
GloVe head	+	-	+	-	-	-	+	-	+	+
GloVe span	-	+	+	-	-	-	+	-	+	+
BERT head	-	-	-	+	-	+	+	+	+	+
BERT span	-	-	-	-	+	+	+	+	+	+
Numberbatch	-	-	-	-	-	-	-	+	+	+
Masked LM	-	-	-	-	-	-	-	-	+	+
Distance	-	-	-	-	-	-	-	-	+	+
Speaker	-	-	-	-	-	-	-	-	+	+
Position	-	-	-	-	-	-	-	-	-	+
CoNLL score:	65.09	62.79	66.28	64.82	66.43	66.59	66.96	68.08	68.13	70.09

Table 4: Average CoNLL scores for different combinations of embeddings in WCS

Table 4 shows some of the results. GloVe head embedding works better than GloVe span embedding (experiments 1 and 2); the difference in CoNLL score is 2.3% (65.09% vs. 62.79%). Combined head and span embeddings (3) further improve the performance by 1.2%. Interestingly, the trend is different for BERT: BERT head (4) achieves 64.82% and BERT span (5) 66.43%. Note that in our case BERT span refers to an average over all BERT embeddings of the tokens within the mention span (this is different from SpanBERT). We observe a very small increase in performance when BERT span is combined with BERT head (6), 66.59% CoNLL score; BERT and Glove combined together (7) achieve 66.96%. We also found that WCS performs better when we add Numberbatch embeddings, e.g., BERT with Numberbatch (8) has 68.08% CoNLL score. We found that the combination of GloVe, BERT, Numberbatch, Masked LM, distance and speaker embeddings (9) gives us 68.13% on the test data. Furthermore, when the position embedding is added (10) the score increases to 70.09 that indicates that the positions of mentions within the document[8] play a very important role in resolving coreference.

Although we tested various combinations of embeddings, our experiments were limited to a subset of the data (we used only 50 documents for training). Another limitation of our approach is that we defined the dimensionality of each embedding separately, i.e., when passing an embedding through several linear layers we defined the output dimensionality independently for each embedding type. It would be interesting to see how interactions between different embeddings affect the dimensionality. For example, it might be that 100 dimensions work well for GloVe when it is used alone but when it is combined with BERT the dimensionality should be increased (or decreased) to achieve better performance. We would also like to test whether ablation studies show similar trends with different types of data (e.g., dialogue vs. non-dialogue, different genres) and perform a proper statistical analysis of the results. We see our present results as a starting point of an investigation of interactions between the embedding types in dialogue, to determine what embeddings work best in which case and what dimensionality they should have.

[8]Position embedding for each mention is computed as a start position divided by the total number of tokens in the document. When computing a score for clustering we concatenate both referent and antecedent positions and then pass them through a linear layer that increases the dimensionality of embedding to 200.