

---

# Edina: Building an Open Domain Socialbot with Self-dialogues

---

Ben Krause Marco Damonte\* Mihai Dobre\* Daniel Duma\* Joachim Fainberg\*  
Federico Fancellu\*† Emmanuel Kahembwe\* Jianpeng Cheng Bonnie Webber‡

School of Informatics  
University of Edinburgh  
Edinburgh, UK

ben.krause@ed.ac.uk, f.fancellu@sms.ed.ac.uk, bonnie@inf.ed.ac.uk

## Abstract

*Edina* is a conversational agent whose responses utilize data harvested from Amazon Mechanical Turk (AMT) through an innovative new technique we call *self-dialogues*. These are conversations in which a single AMT Worker plays both participants in a dialogue. Such dialogues are surprisingly natural, efficient to collect and reflective of relevant and/or trending topics. These self-dialogues provide training data for a generative neural network as well as a basis for *soft rules* used by a matching score component. We present methodology for combining rule-based, retrieval, and generative methods to effectively leverage our data. Our hybrid data-driven methodology thus addresses both coverage limitations of a strictly rule-based approach and the lack of guarantees of a strictly machine-learning approach.

## 1 Introduction

Building an open-domain conversational AI for commercial use poses two main challenges. First is broad-coverage: Modeling natural conversation in an unrestricted number of topics is still an open problem as shown by the current concentration of research on dialogue in restricted domains (e.g. Bowden and et al. [2017]). Second is the scarcity of clean, unbiased and comprehensive datasets of open-ended conversation, which makes it difficult to develop conversational dialogue systems and limits the viability of using purely data-driven methods (e.g. neural networks).

Our approach aims to integrate rule-based with machine-learned behavior, with both grounded in data that is *intelligible*, *reflective* of what people want to talk about, and *gathered automatically*. This means acknowledging that people talk differently about different things at different times. The domain has to be potentially unrestricted and the model should be powerful enough to ensure that the most likely system response is in line with both the customer’s most recent utterance and, ideally, the flow of the overall conversation. But most importantly, we want to keep the customer — Alexa’s conversational partner — in the loop since Alexa’s main purpose is to keep the customer engaged.

We present *Edina*, a conversational AI agent that exploits a corpus of conversations harvested from Amazon Mechanical Turk (§2). Our primary innovation is to collect and use data in the form of **self-dialogues**, in which Workers engage in conversation with themselves on a specific topic, enabling us to gather first-hand what people discuss and how they discuss it. By running tasks periodically, we are able to identify the trending entities people talk about at a given point in time. Although our system architecture (§3) also includes a rule-based component (§3.1), our growing corpus of self-dialogues is designed to enable us to avoid having to hand-script potential conversational situations. We achieve this by an idf (inverse document frequency)-based matching score (§3.3) that takes a user utterance and returns the most likely response, based on either conversational partner in a self-dialogue. We also develop a specific retrieval formula for conversational agents, and demonstrate a novel way

---

\* equal contribution; † team leader; ‡ faculty advisor

of integrating our retrieval method with other components via a confidence score. Although *Edina* currently converses on only three main topics (movies, music and sports), the approach is **easily extensible** to any topic. Our approach is also **cost-effective**: Once an initial quantity of data is collected around a topic, only small updates are needed in order to stay current on trending topics. At the end of the paper, alongside the discussion (§5), we describe two further improvements that we plan to implement in future work.

## 2 Data collection

Our focus on data collection stems from the scarcity of publicly available corpora for training dialogue systems. During the first stage of the competition, we surveyed the corpora mentioned in Serban et al. [2015] and found that suitable publicly available corpora were typically too small, artificial or difficult to obtain. We instead chose to gather our own data with the primary criteria that the conversations i) represent casual human dialogue and ii) have identifiable topics. We turned to Amazon Mechanical Turk (AMT), where we initially developed a chat interface for two AMT Workers to chat about a specific topic. The interface was based on VisDial [Abhishek and et al., 2016], an interface that prompts the Workers to converse about a particular image. However, in demanding that two workers be connected at the same time, this setup slowed down the collection task on an otherwise fast-moving platform. This led us to develop an innovative new task on AMT, **self-dialogue**, in which a single worker is asked to play both parts in a conversation about a given topic. A sample self-dialogue is given in Figure 1, alongside the instructions provided to the AMT Workers. A key requirement is that the conversations appear natural. For this reason, we avoided lengthy instruction, instead allowing Workers to interpret the task as they wish within the given topic.

Overall we observed the following data-related benefits of self-dialogues, as compared with the two-person conversations that are more standardly collected:

1. **Collection speed and efficiency of set up**: Self-dialogues do not require waiting for two workers to connect at the same time, speeding up the collection process and easing the development of the back-end.
2. **Data quality**: Both parties of a self-dialogue are equally expert in the same topics, making it much easier for self-dialogues to go into detail about particular topics. We also found that two-person dialogues often showed misunderstandings between speakers, and clarifications that play out over several turns, requiring undesirable complexity of the conversation agent.
3. **Naturalness**: It was striking the extent to which people were able to have interesting and engaging conversations with themselves, adopting different perspectives and even different stances with respect to a topic, depending on which participant they were pretending to be at each point in the conversation. While one might have expected the conversations to simply resemble question-answer pairs, participants instead often contributed a pair of communicative actions — often with the first part addressing the previous conversation and the other, advancing it further. Such contributions are particularly valuable as data because they have built-in hook to further engagement.
4. **Cost effectiveness**: Related to (2), we were able to halve the cost of collecting conversation since we only had to pay one worker instead of two.

Submissions were of surprisingly high quality. We implemented a pipeline to review and identify Workers abusing the system or submitting duplicates. However, only eight of 2,717 were banned, and only 145 conversations ( $\approx 0.6\%$ ) were rejected. Initial experiments with requirements on Workers and the pay per conversation converged on the following set:

- location: the United States or United Kingdom;
- HIT approval rate: greater than 95%;
- number of HITs approved: greater than 500;
- number of conversations per worker per task: maximum 20;
- pay per 10-turn conversation: US \$0.70-80; 5-turn conversation: US \$0.35-40.

## Task and instructions

Assume there are two users talking to each other about **NFL football**. Your job is to pretend to be both people and hold a conversation:

**Note** - The topic must be "**NFL football**". Keep each turn to 1-2 sentences. **Avoid bad, political, sexist and inappropriate language and avoid IM/chat abbreviations.**

**Example:**

**User1:** Hey are you excited for the upcoming NFL season?

**User2:** Yeah for sure! What team do you think will win it all?

**User1:** I think the patriots are going to repeat.

**User2:** Tom Brady is getting old, I think this year will belong to the Seahawks.

**User1:** I don't know about that, the Seahawks haven't been the same without Marshawn Lynch.

**[Including some facts in the conversations are a great touch!]**

etc...

**Avoid bad, political, sexist and inappropriate language.**

Now it's your turn. Fill in for both users for 10 turns. Incomplete submissions will be rejected. You can do up to 20 of these; though obvious copies and exploitations will be rejected.

Figure 1: AMT interface for the NFL Football task.

Workers produced 10-turn dialogues for the main topics, and 5-turn dialogues for sub-topics (except for "NFL Football"). Empirically, we observed that the 10-turn tasks afforded Workers the time to transition into a topic of interest, which was unnecessary for sub-topics.

To date, we have collected 24,283 self-dialogues through AMT, across four major topics separated into 23 tasks. These currently amount to 3,653,313 words, across 141,945 turns, from 2,717 Workers. On average, each worker has submitted  $\sim 9$  self-dialogues, so the corpus displays a wide variety of styles, content and ideas across a large population.

The total paid to Workers so far is US \$17,947.54. AMT's high liquidity has enabled us to gather data fast when required. In a period of approximately 20 days (from 20/06 to 10/07) we were able to gather about 20,000 conversations, with a peak of 2,307 conversations collected in a single day. The following is an excerpt from a self-dialogue in the Movies category:

What is your absolute favorite movie?  
I think Beauty and the Beast is my favorite.  
The new one?  
No, the cartoon. Something about it just feels magical.  
It is my favorite Disney movie.  
What's your favorite movie in general?  
I think my favorite is The Sound of Music.  
Really? Other than cartoons and stuff I can never get into musicals.  
I love musicals. I really liked Phantom of the Opera.

The data (or *offline conversations*) are stored by grouping them by their corresponding tasks, with each line in each conversation as a response. In Section 3.3, we explain how a context is associated with each response for use with the Matching Score component. We propose additional use of the data for improving the agent in Section 5.

### 3 System Architecture

Our system is a hybrid of several rule-based and data-driven components. Following an initial preprocessing step, our system exploits a priority queue of components that vary in their intended coverage and functionality. Its three **main** components comprise:

1. A **rule-based** component that uses a list of hand written rules and templates. It returns a response only if the user's input is an exact match. While this component has the highest

Topic/subtopic	# Conversations	# Words	# Turns
Movies	4,126	814,842	82,018
Action	414	37,037	4,140
Comedy	414	36,401	4,140
Fast & Furious	343	33,964	3,430
Harry Potter	414	44,220	4,140
Disney	2,331	232,573	23,287
Horror	414	428,33	4,138
Thriller	828	77,975	8,277
Star Wars	1,726	178,351	17,260
Superhero	414	40,967	4,140
Music	4,911	924,993	98,123
Pop	684	62,383	6,840
Rap / Hip-Hop	684	66,376	6,840
Rock	684	63,349	6,837
The Beatles	679	68,396	6,781
Lady Gaga	558	49,313	5,566
Music and Movies	216	37,303	4,320
NFL Football	2,801	562,801	55,939

Table 1: Data collected on AMT and used for Edina. Each line is an individual task, i.e. *Movies* is not a combination of the related subcategories.

priority, due to obvious coverage limitations, it often returns nothing and defers to the matching score component.

2. A **matching score** component that selects responses from a pool of conversational data, based on how close the context of the user conversation is to the context of the response in our data. The matching score also returns a confidence score that is used to better control its interaction with the other components.
3. A **generative neural network** that always generates a response and is deployed if the other two components fail. It often gives general and vague on topic responses as compared with the more specific responses of the matching score.

Interleaved between these three main components are three *minor* components:

4. **EVI**, which serves as our primary IR engine, but is only used to reply to customer inputs that do not probe the system’s identity, personality or opinions. EVI takes priority in responding to WH-questions questions that require an exhaustive knowledge base. As conversations to date have rarely taken this route, EVI has rarely been called.
5. A **likes and dislikes** component whose purpose is to answer questions about Edina’s opinions on entities and definitions that the rule-based component fails to cover.
6. A **proactive component** which asks the customer a question or series of questions, in order to steer the conversation back to what the matching score can handle. We limit its use, in order to avoid probing the customer too often, resulting in an unpleasant experience.

Although the system is guaranteed to always return something, we retain a set of interesting facts or clarification replies in the unlikely case that all components fail — say in case of major network errors or if the output from all other components is judged offensive. Output is always filtered for the possibility of profanity, and we have extended the list of profanities provided by Amazon with our own additions. Table 2 shows how often each components’ output is chosen to be sent back to the Alexa device. Given the order in our priority queue, one can observe the challenges of creating a purely rule-based system that can guarantee full coverage. The rule-based component can only reply to  $\approx 16\%$  of the customers’ utterances. Our agent overcomes this limitation by employing data driven methods, i.e. the matching score (§3.3) and generative neural network (§3.6) components, that make for the majority of the system’s replies ( $\approx 67\%$ ).

Figure 2 presents an overview of the complete system, from input to the Alexa device to the output generated by *Edina* (i.e. green arrow). To overcome high traffic and prevent local failures, an Elastic

Load Balancer (ELB) redirects the traffic to two instances of *Edina* (i.e. CORE block). In addition, all components, including the core, are multi-threaded and can automatically restart any threads that fail. Components either have a single dedicated EC2 instance or share one, depending on their computational requirements. The components in each of the blocks run in parallel. The previously described priority queue is applied once all the components return an output or timeout. Figure 3 shows the average latency reported by the load balancer. The system’s average reply is well under 1 second, while bounding the maximum to a reasonable delay in the periods with the highest traffic.

*Edina*’s conversations with Amazon customers as well as other information related to the conversation (e.g. topic, customer preferences, the outputs from each of our components etc.) are stored in a Postgres Amazon RDS instance. Throughout the document, we refer to dialogues with Amazon customers as *online conversations* in order to differentiate them from the self-dialogues collected via Amazon Mechanical Turk (*offline conversations*).

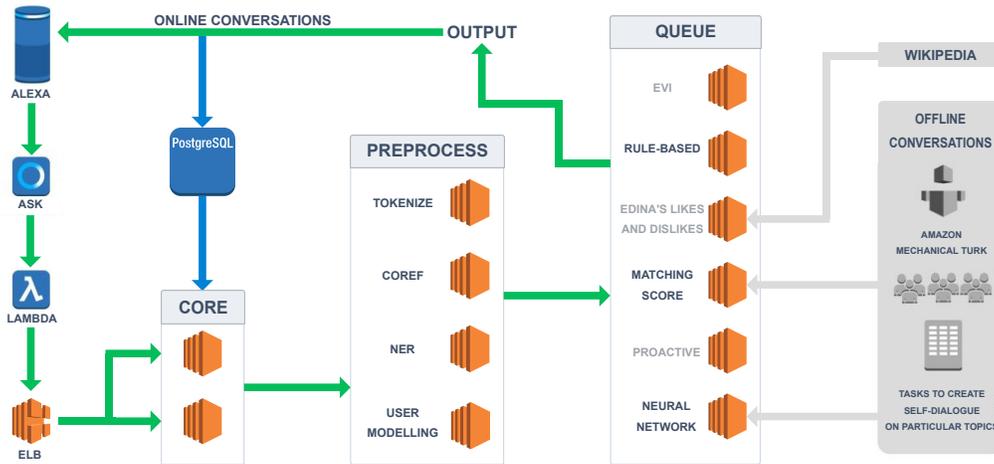


Figure 2: Topological system diagram that shows the order of execution.

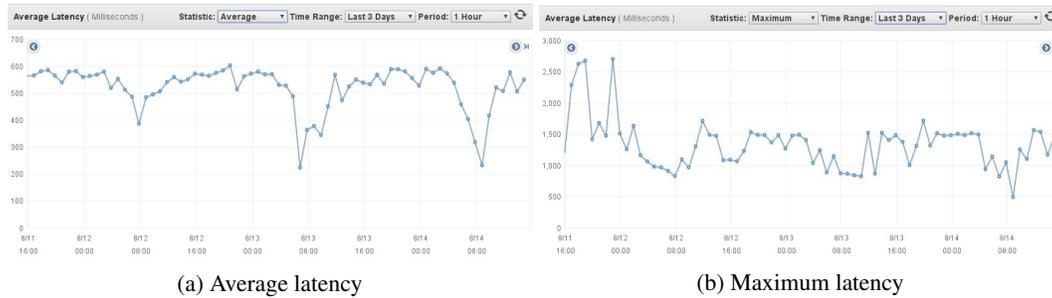


Figure 3: Latency experienced by the elastic load balancer.

RB	MS	NN	EVI	LD	Proactive	Backup
15.69%	46.29%	20.55%	0.47%	0.37%	16.56%	0.08%

Table 2: Usage frequency of each component. RB is the rule-based, MS is the matching score, NN is the generative neural network, LD is the likes and dislikes, and Backup is last resort trivia.

### 3.1 Preprocessing

We perform a series of preprocessing steps on the raw input received from Lambda. The input is first processed through spaCy’s<sup>1</sup> pipeline mainly for tokenization and Named Entity Recognition (NER). Remaining preprocessing is performed in parallel, comprising another NER, coreference resolution and simple user-modeling. The second NER is performed using DBpedia Spotlight API [Daiber et al., 2013] which extracts more information on the named entities from Wikipedia. We perform the coreference annotation (COREF) over the previous four turns (i.e. 2 turns each from the user and the bot), using Stanford CoreNLP deterministic implementation [Recasens et al., 2013]. The generated coreference chain is used to modify the current input message by replacing pronouns with the entities they refer to. Finally, user-modeling is a simple rule-based implementation that catches explicit mentions of the user’s likes and dislikes, as well as when the user would like to change the topic. These preferences are matched to a list of topics and subtopics that our system can handle. The processed input and the additional information that result from the preprocessing phase is sent to each of the components that build a reply.

### 3.2 Rule-based Component

The rule-based component deterministically matches customer input and returns a single output. Rules in the current ruleset address the following points:

- **the agent’s identity and preferences.** Rules ensure that identity information remains anonymous for the moment (e.g. name and location). Most preferences are personalized so to make the conversational agent more human-like.
- **sensitive topics** such as suicide, cancer or death of a close person, which we carefully redirect to existing helplines when possible. Prompts containing a list of sensitive and/or potentially offensive words are also handled by a polite yet firm response (e.g. ‘This kind of talk makes me uncomfortable, let’s talk about something else.’).
- **topic shifting**, so as to recognize when the customer wants to set a new topic or change the current one, or when the agent should shift away from controversial topics, such as politics, which we do not handle.
- **other forms of engagement.** These enable the agent to make jokes, to play a small point-based game, where the customer has to complete the lyrics of popular songs, or to invoke a weather API that can return information about the weather, given the customer’s location.

Although the development of the rule-based component was based solely on intuition about what the rules should capture, we have continued to refine our rules and add to our ruleset, based on conversations between Alexa users and *Edina*. Finally, we also integrated the list of most common customer prompts provided by Amazon. The code has been implemented in RiveScript<sup>2</sup>.

### 3.3 Matching Score component

Given a customer utterance  $q$ , the matching score component is designed to return the most appropriate response  $r$  from the bank of self-dialogues (Section §2). We treat this bank  $B$  as a set of tuples  $\{b_1, b_2, \dots, b_n\}$  where  $b_i$  is a tuple  $(r_i, c_i)$  containing a response  $r_i$  and a context  $c_i$ , and  $c_i$  is either the immediately preceding response (hence,  $c_i=r_{i-1}$ ) or the one at one remove ( $c_i=r_{i-2}$ ). In the context of an online conversation, we define as  $q_c$  as the response from *Edina* directly preceding the utterance  $q$ . This notation is illustrated below:

#### Online conversation

previous response from *Edina* ( $q_c$ ): what’s your favorite movie

customer utterance ( $q$ ): sound of music

#### Off-line Conversation (from $B$ )

$r_{i-2}$ : What’s your favorite movie in general?

$r_{i-1}$ : I think my favorite is The Sound of Music.

$r_i$ : Really? Other than cartoons and stuff I can never get into musicals.

(where  $b_i = (r_i, c_i)$  and  $c_i = r_{i-1} \vee r_{i-2}$ )

<sup>1</sup><https://www.spacy.io>

<sup>2</sup><https://www.rivescript.com>

In order to retrieve the most likely response from  $B$ , we use a scoring function  $\mathcal{S}(q, r_i, c_i)$  that measures the similarity between the customer utterance  $q$  and the response  $r_i$  with a context  $c_i$ . This similarity measure is based on bag-of-words vectors which up-weight rare words using inverse document frequency (IDF).<sup>3</sup> Using bag-of-words permits inverted indexing, where response IDs are stored in a look-up table of words. This removes the need to directly compare  $q$  with every  $r_i$ .

More formally we define our final matching score  $\mathcal{S}(q, r_i, c_i)$  as an interpolation of three different subscores plus a normalization term  $\eta$ :

$$\mathcal{S}(q, r_i, c_i) = \frac{(S^c + S^{cr})(S^c)^n}{\eta} + \lambda S^{2cq} \quad (1)$$

where  $\lambda$  and  $n$  are constant ( $\lambda = 0.005$  and  $n = 0.5$ ).

$S_i^c$  is the most important term and measures the similarity between  $q$  and  $c_i$ . Here the context is limited to one preceding response, hence  $c_i = \{r_{i-1}\}$ . Instead of normal IDF scores, we take their cube to give rarer words a higher relative weight. Formally,  $S_i^c$  can be defined as:

$$S^c = q^3 \cdot c_i^3 \quad (2)$$

Using this inner product term alone, however, would result in (1) a matching score with a strong preference for responses with high  $\|c_i^3\|_2$ , and (2) higher scores for all matches where  $\|q^3\|_2$  is high. Therefore we introduce a normalization term

$$\eta = \|q^3\|_2 \|c_i^3\|_2 \quad (3)$$

Dividing  $S_i^c$  by  $\eta$  would yield a cosine similarity based score.

One shortcoming of  $S_i^c$  is that we might not want to perform exact match on very generic context such as simply "yes". In this case, it might be better for the matching score to defer to another component rather than find a match for "yes" in  $B$ . It might also be desirable to consider the response  $r_i$  when up-weighting overlapping words, because responses that contain words from the query are often more engaging and sometimes more relevant.

$S_i^{cr}$  addresses both issues by i) computing the inner product of the quartic of the IDF scores so to create even more contrast between common and rare words and ii) calculating the similarity between the query and the response. This is formalized as:

$$S^{cr} = q^4 \cdot r_i^4 \quad (4)$$

It can also be useful to consider a wider context  $c_i$  as well as more than one previous customer utterances when choosing a response.

$S^{2cq}$  does that by considering as context  $r_{i-2}$  as well as the agent response preceding  $q_c$ . For this subscore, normal IDF scores are used. This translates into the following equation:

$$S^{2cq} = q_c \cdot c_i, \text{ where } c_i = r_{i-2} \quad (5)$$

In the case of ties or near ties, we select the argmax of  $\|r_i\|_2$  to try to select a response that is longer and with more rare words.

Finally, the matching score keeps track of the most recent topic or subtopic. When a response is pulled from a particular sub-dataset from our AMT data, the sub-dataset topic is assumed to be the current topic. All responses from the sub-dataset covering the current topic are given a small bonus to their score to improve topic coherence. The matching score returns the highest scoring response as well as the score of that response, which we refer to as the confidence score.

### 3.4 Likes and dislikes

This component is triggered only if the rule-based agent fails to return and the matching score confidence drops below a predefined threshold. It contains a set of very specific templates that depend on the number of recent views of the entity's Wikipedia page. These templates are limited to 'safe' domains, e.g. actors, singers and bands, to avoid sensitive entities. It is particularly useful for unseen entities. An example response by this component is shown below in (1).

- (1) Customer: Do you like Kaizers Orchestra?  
Edina: Kaizers Orchestra was a Norwegian alternative rock band formed on 1 January 2000. What do you think about them?

<sup>3</sup>Word frequencies for IDF scores were taken from [http://norvig.com/ngrams/count\\_1w.txt](http://norvig.com/ngrams/count_1w.txt).

### 3.5 Proactive component

The proactive component is designed to drive the conversation towards a state that other components are better able to handle. It does so by sampling from a set of questions to ask the customer. These questions are designed to either:

- probe the customer to mention a specific entity that the Matching Score component will likely be able to match;
- ask a yes or no question to gauge the customer’s interest in a particular topic, followed by a question probing the customer to continue the conversation within that topic (cf. Example 2).

The proactive component is called when the matching score fails to return a response due to low confidence. It is in general beneficial if we can probe the user to mention specific entities related to the topics in our data, as our Matching Score component can often give interesting opinions about specific entities. The proactive component is limited to only returning a response once every 5 turns to avoid excessive use. An example interaction is shown below.

- (2) Edina: Did you enjoy the last Star Wars movie?  
Customer: Yes  
Edina: I was hoping for you to say yes! What did you think of it?

### 3.6 Generative Neural Network component

The generative neural network is a recurrent neural network (RNN) language model that can predict a probability over the next word given a history of words. For this component, we used a multiplicative LSTM [Krause et al., 2017] which combines features of a multiplicative RNN [Sutskever et al., 2011] and an LSTM [Hochreiter and Schmidhuber, 1997] to achieve a stronger language modeling performance. We pretrained our network on the open subtitles dataset [Lison and Tiedemann, 2016] using vocabulary from our self-dialogues. We then fine-tuned this model on our self-dialogue data sources. We considered several stopping points to try to obtain the model that generates the best samples. In general, some degree of overfitting (continuing to train even after the validation error is getting worse) was beneficial to sample quality. At runtime, we simply sampled this model word by word, conditioned on the customer’s last response, and the agent’s response before that. We used a temperature of 0.7 for sampling, to draw samples from higher density regions under the model.

## 4 System Evaluation

### 4.1 Quantitative evaluation of matching score

We evaluated our matching score component and its associated confidence scores using an internal human evaluation metric. A single trial in our evaluation process involved sampling a triple  $(r_{i-2}, r_{i-1}, r_i)$  from our data-bank of self-dialogues. We then use this conversational context as the user utterance for our matching score ( $q_c = r_{i-2}$  and  $q = r_{i-1}$ ). We then find the top 4 responses under our scoring function  $\mathcal{S}(q, r_j, c_j)$ , for  $i \neq j$ , forcing the matching score to chose a response other than the true response  $r_i$  which was given by the AMT worker. We randomly sampled response  $r$  to either be one of these top-4 matching score responses, or the response given by the AMT worker ( $r = r_i$ ). The response  $r$  was then rated in the context of  $q_c$  and  $q$  according to the evaluation metric given below. The rater was always blind to whether  $r$  was a real AMT worker response.

#### Evaluation scale

5 - human-like response that is interesting or engaging

4 - human-like response that is not as interesting or engaging, or response that doesn’t quite fit conversationally but is interesting or engaging

3 - probably not a human-like response, but still somehow addresses part of the conversation context

2 - something is clearly off about response, but still could contribute to the conversation in some way

1 - response is incoherent.

We collected 181 trials of human evaluations of top-4 matching score responses and 33 trials of human evaluations of real responses from AMT workers. We found that the confidence scores of our top-4 matching score responses were positively correlated with human evaluation scores ( $r = 0.259$ ,

Response type	Mean human evaluation score
Human AMT worker	4.09 ± 0.15
Matching score top 4, confidence > 0.7	3.60 ± 0.07
Matching score top-4, confidence > 0.5	3.50 ± 0.08
Matching score top-4	3.08 ± 0.09

Table 3: Average human evaluation of matching score and real human responses, plus or minus standard error. Confidence threshold of 0.5 corresponded to confidence threshold for matching score component to response. Confidence threshold of 0.7 corresponded to confidence threshold for matching score component to move ahead in the queue of the likes and dislikes component.

$p < 0.0005$ ). Statistics on human evaluation of matching score at different confidence thresholds and real human responses are given in Table 3.

These experiments demonstrate the utility of applying a confidence score to our retrieval based method. When the confidence score is high, the response quality is generally strong. However, when the confidence score is low, the matching score can self-censor, allowing another more general component to respond. The ability to filter nonsensical responses is critical for integrating data driven components into conversational agents.

## 4.2 Qualitative evaluation of user conversations

Following, we show an excerpt from a conversation that a member of our team had with the bot.

Bot: What bands have you seen live?  
 User: I've seen Sufjan Stevens, do you like them?  
 Bot: What's he like?  
 User: Quite boring to be honest  
 Bot: I know what you are talking about .  
 User: Right?! What kind of **bands** have you seen live?  
 Bot: Yeah! I saw **Cake** a few months ago, and just last week I saw **The Protomen**.  
 User: Really! How was Cake?  
 Bot: Eating cake?  
 User: No I'm not eating a cake  
 Bot: No, the band. Have you ever heard of **them**?  
 User: No i haven't  
 Bot: **She** does more traditional type music, but I absolutely love the way **she** sings.  
 User: I'll have to check her out  
 Bot: You should! she's the best singer for sure .

Gathering data from potential users allows for the bot to converse about specific named entities (as highlighted in **blue**) and to generate fully intelligible output. The flow of the conversation around a particular topic is also natural with the bot being able to correct itself when misinterpreting an ambiguous term ('cake'(food) vs. 'Cake' (band)). However, coreference is still an issue, as evidenced by some pronoun mismatches (as highlighted in **red**).

## 5 Discussion and future work

*Edina* demonstrated the ability to engage in a wide range of subtopics during conversations with Alexa customers. This was achieved by drawing responses from a conversation dataset created with the novel technique of *self-dialogues*. This technique plays a significant role in enabling our agent to generate relevant, topical output closely resembling human conversations. Our internal system evaluation further demonstrated the value of using a thresholded matching score to select a response. During our human evaluation, the responses that have a high matching confidence score received a significantly higher rating compared to the other responses (see Table 3), showing that the matching score can self-censor to greatly reduce the risk of a poor response.

We believe the strength of our system is to incorporate the advantages of both data-driven and rule-based approaches while avoiding their shortcomings. We were able to cover a broad range of

topic and subtopics without the need of tailored scripts or rules, while at the same time using data that does not contain a large amount of noise and is easy to collect. Our self-conversation data collection technique and our unique approach to integrating rule-based, retrieval, and machine learning methods should prove useful for future open domain conversational agents.

For future work we are planning to focus on the following two points:

1. **Improving entity tracking:** our co-reference system is not always effective in a conversational context. For instance, when asked “how old is that movie?” in reference to a movie starring Johnny Depp the matching score component returned “It’s from 1954, is that too old?”. A solution to this could be to store a list of the most common entities in our dataset and keep track of these when used in our online conversations. This list can also be coupled with the current conversation topic and reset when the topic changes.
2. **Self-dialogue as direct supervision signal:** the task of self-dialogue can also be used directly improve *Edina*’s response. More specifically, we are planning an AMT task where workers would converse with *Edina* and each time the conversational initiative returns to the Worker, she or he could choose to replace the agent’s response with an alternative, if *Edina*’s initial response was not satisfactory. When a Worker replaces *Edina*’s response, the worker would then respond to their own response, and the agent would continue the conversation at its next turn. The alternatives proposed by the Worker would be indexed by the matching score, so that if the agent encounters the same situation in a future conversation, it will give the Worker’s proposed response. Given that we showed the effectiveness of self-dialogues and our matching score, we believe this method would be very effective at teaching our agent to give good responses in situations where it is currently weak.

## References

- D. Abhishek and S. Kottur et al. Visual dialog. *CoRR*, abs/1611.08669, 2016. URL <http://arxiv.org/abs/1611.08669>.
- K. Bowden and S. Oraby et al. Data-driven dialogue systems for social agents. In *International Workshop on Spoken Dialogue Systems*, 2017.
- Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*, 2013.
- Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.
- B. Krause, I. Murray, S. Renals, and L. Lu. Multiplicative LSTM for sequence modelling. *ICLR Workshop track*, 2017. URL <https://openreview.net/forum?id=SJCS5rXF1>.
- Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, May 2016. European Language Resources Association.
- Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. The life and death of discourse entities: Identifying singleton mentions. In *North American Association for Computational Linguistics (NAACL)*, 2013.
- Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. A survey of available corpora for building data-driven dialogue systems. *CoRR*, abs/1512.05742, 2015. URL <http://arxiv.org/abs/1512.05742>.
- I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.