Neural Generation of Dialogue Response Timings

Matthew Roddy and Naomi Harte
ADAPT Centre, School of Engineering
Trinity College Dublin, Ireland
{roddym,nharte}@tcd.ie

Abstract

The timings of spoken response offsets in human dialogue have been shown to vary based on contextual elements of the dialogue. We propose neural models that simulate the distributions of these response offsets, taking into account the response turn as well as the preceding turn. The models are designed to be integrated into the pipeline of an incremental spoken dialogue system (SDS). We evaluate our models using offline experiments as well as human listening tests. We show that human listeners consider certain response timings to be more natural based on the dialogue context. The introduction of these models into SDS pipelines could increase the perceived naturalness of interactions.[1]

1 Introduction

The components needed for the design of spoken dialogue systems (SDSs) that can communicate in a realistic human fashion have seen rapid advancements in recent years (e.g. Li et al. (2016); Zhou et al. (2018); Skerry-Ryan et al. (2018)). However, an element of natural spoken conversation that is often overlooked in SDS design is the timing of system responses. Many turn-taking components for SDSs are designed with the objective of avoiding interrupting the user while keeping the lengths of gaps and overlaps as low as possible e.g. Raux and Eskenazi (2009). This approach does not emulate naturalistic response offsets, since in human-human conversation the distributions of response timing offsets have been shown to differ based on the context of the first speaker's turn and the context of the addressee's response (Sacks et al., 1974; Levinson and Torreira, 2015; Heeman and Lunsford, 2017). It has also been shown that listeners have different anticipations about upcoming

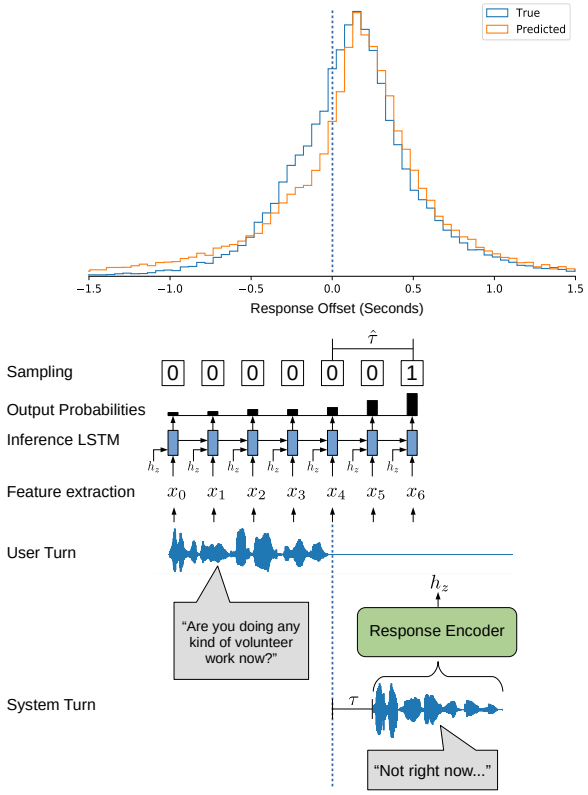


Figure 1: Overview of how our model generates the distribution of turn-switch offset timings using an encoding of a dialogue system response h_z, and features extracted from the user's speech x_n.

responses based on the length of a silence before a response (Bögels et al., 2019). If we wish to realistically generate offsets distributions in SDSs, we need to design response timing models that take into account the context of the user's speech and the upcoming system response. For example, offsets where the first speaker's turn is a *backchannel* occur in overlap more frequently (Levinson and Torreira, 2015). It has also been observed that *dispreferred* responses (responses that are not in line with the suggested action in the prior turn) are associated with longer delays (Kendrick and Torreira, 2015; Bögels et al., 2019).

[1] Our code is available at https://github.com/mattroddy/RTNets.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2442–2452
July 5 - 10, 2020.

Overview We propose a neural model for generating these response timings in SDSs (shown in Fig. 1). The response timing network (RTNet) operates using both acoustic and linguistic features extracted from user and system turns. The two main components are an encoder, which encodes the system response h_z, and an inference network, which takes a concatenation of user features (x_n) and h_z. RTNet operates within an incremental SDS framework (Schlangen and Skantze, 2011) where information about upcoming system responses may be available before the user has finished speaking. RTNet also functions independently of higher-level turn-taking decisions that are traditionally made by the dialogue manager (DM) component. Typically, the DM decides when the system should take a turn and also supplies the natural language generation (NLG) component with a semantic representation of the system response (e.g. intents, dialogue acts, or an equivalent neural representation). Any of the system response representations that are downstream from the DM's output representation (e.g. lexical or acoustic features) can potentially be used to generate the response encoding. Therefore, we assume that the decision for the system to take a turn has already been made by the DM and our objective is to predict (on a frame-by-frame basis) the appropriate time to trigger the system turn.

It may be impractical in an incremental framework to generate a full system response and then re-encode it using the response encoder of RTNet. To address this issue, we propose an extension of RTNet that uses a variational autoencoder (VAE) (Kingma and Welling, 2014) to train an interpretable latent space which can be used to bypass the encoding process at inference-time. This extension (RTNet-VAE) allows the benefit of having a data-driven neural representation of response encodings that can be manipulated without the overhead of the encoding process. This representation can be manipulated using vector algebra in a flexible manner by the DM to generate appropriate timings for a given response.

Our model's architecture is similar to VAEs with recurrent encoders and decoders proposed in Bowman et al. (2016); Ha and Eck (2018); Roberts et al. (2018). Our use of a VAE to cluster dialogue acts is similar to the approach used in Zhao et al. (2017). Our vector-based representation of dialogue acts takes inspiration from the 'attribute vectors' used in Roberts et al. (2018) for learning musical structure representations. Our model is also related to continuous turn-taking systems (Skantze, 2017) in that our model is trained to predict future speech behavior on a frame-by-frame basis. The encoder uses a multiscale RNN architecture similar to the one proposed in Roddy et al. (2018) to fuse information across modalities. Models that intentionally generate responsive overlap have been proposed in DeVault et al. (2011); Dethlefs et al. (2012). While other models have also been proposed that generate appropriate response timings for fillers (Nakanishi et al., 2018; Lala et al., 2019) and backchannels (Morency et al., 2010; Meena et al., 2014; Lala et al., 2017).

This paper is structured as follows: First, we present how our dataset is structured and our training objective. Then, in sections 2.1 and 2.2 we present details of our two models, RTNet and RTNet-VAE. Section 2.3 presents our input feature representations. In section 2.4 we discuss our training and testing procedures. In sections 3.1 and 3.2 we analyze the performance of both RTNet and RTNet-VAE. Finally, in section 4 we present the results of a human listener test.

2 Methodology

Dataset Our dataset is extracted from the Switchboard-1 Release 2 corpus (Godfrey and Holliman, 1997). Switchboard has 2438 dyadic telephone conversations with a total length of approximately 260 hours. The dataset consists of pairs of adjacent turns by different speakers which we refer to as *turn pairs* (shown in Fig. 2). Turn pairs are automatically extracted from orthographic annotations using the following procedure: We extract frame-based speech-activity labels for each speaker using a frame step-size of 50ms. The frame-based representation is used to partition each person's speech signal into *interpausal units* (IPUs). We define IPUs as segments of speech by a person that are separated by pauses of 200ms or greater. IPUs are then used to automatically extract *turns*, which we define as consecutive IPUs by a speaker in which there is no speech by the other speaker in the silence between the IPUs. A turn pair is then defined as being any two adjacent turns by different speakers. The earlier of the two turns in a pair is considered to be the *user turn* and the second is considered to be the *system turn*.

Training Objective Our training objective is to predict the start of the system turn one frame ahead

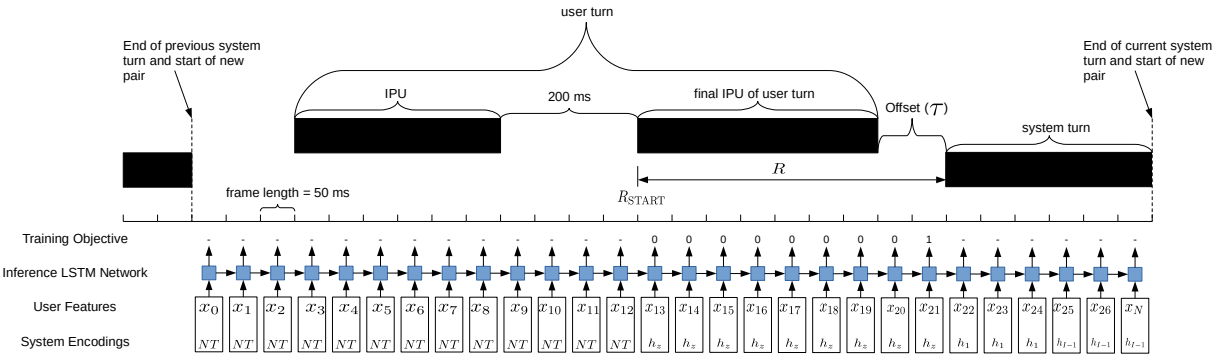


Figure 2: Segmentation of data into *turn pairs*, and how the inference LSTM makes predictions.

of the ground truth start time. The target labels in each turn pair are derived from the ground truth speech activity labels as shown in Fig. 2. Each 50 ms frame has a label $y \in \{0, 1\}$, which consists of the ground truth voice activity shifted to the left by one frame. As shown in the figure, we only include frames in the span R in our training loss. We define the span R as the frames from the beginning of the last IPU in the user turn to the frame immediately prior to the start of the system turn.

We do not predict at earlier frames since we assume that at these mid-turn-pauses the DM has not decided to take a turn yet, either because it expects the user to continue, or it has not formulated one yet. As mentioned previously in section 1, we design RTNet to be abstracted from the turn-taking decisions themselves. If we were to include pauses prior to the turn-final silence, our response generation system would be additionally burdened with making turn-taking decisions, namely, classifying between mid-turn-pauses and end-of-turn silences. We therefore make the modelling assumption that the system's response is formulated at some point during the user's turn-final IPU. To simulate this assumption we sample an index R_{START} from the span of R using a uniform distribution. We then use the reduced set of frames from R_{START} to R_{END} in the calculation of our loss.

2.1 Response Timing Network (RTNet)

Encoder The encoder of RTNet (shown in Fig. 3) fuses the acoustic and linguistic modalities from a system response using three bi-directional LSTMs. Each modality is processed at independent timescales and then fused in a master Bi-LSTM which operates at the linguistic temporal rate. The output of the master Bi-LSTM is a sequence of encodings $h_0, h_1, ...h_I$, where each encoding is a concatenation of the forward and backward hidden states of the master Bi-LSTM at each word index.

The linguistic Bi-LSTM takes as input the sequence of 300-dimensional embeddings of the tokenized system response. We use three special tokens: SIL, WAIT, and NONE. The SIL token is used whenever there is a gap between words that is greater than the frame-size (50ms). The WAIT and NONE tokens are inserted as the first and last tokens of the system response sequence respectively. The concatenation $[h_0; h_1; h_I]$ is passed as input to a RELU layer (we refer to this layer as the *reduction layer*) which outputs the h_z encoding. The h_z encoding is used (along with user features) in the concatenated input to the inference network. Since the WAIT embedding corresponds to the h_0 output of the master Bi-LSTM and the NONE embedding corresponds to h_I, the two embeddings serve as "triggering" symbols that allow the linguistic and master Bi-LSTM to output relevant information accumulated in their cell states.

The acoustic Bi-LSTM takes as input the sequence of acoustic features and outputs a sequence of hidden states at every 50ms frame. As shown in Fig. 3, we select the acoustic hidden states that correspond to the starting frame of each linguistic token and concatenate them with the linguistic hidden states. Since there are no acoustic features available for the WAIT and NONE tokens, we train two embeddings to replace these acoustic LSTM states (shown in purple in Fig. 3). The use of acoustic embeddings results in there being no connection between the WAIT acoustic embedding and the first acoustic hidden state. For this reason we include h_1 in the $[h_0; h_1; h_I]$ concatenation, in order to make it easier for information captured by the the acoustic bi-LSTM to be passed through to the final concatenation.

Inference Network The aim of our inference network is to predict a sequence of output probabilities $Y = [y_{R_{\text{START}}}, y_{R_{\text{START}}+1}, ..., y_N]$ using

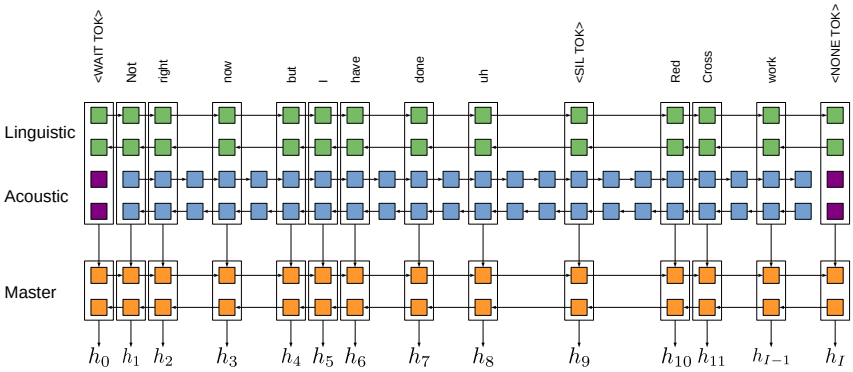


Figure 3: The encoder is three stacked Bi-LSTMs. We use special embeddings (shown in purple) to represent the acoustic states corresponding to the first and last tokens (WAIT and NONE) of the system's turn.

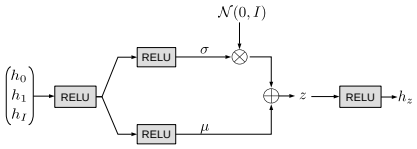


Figure 4: VAE

a response encoding h_z, and a sequence of user features $X = [x_0, x_1, ..., x_N]$. We use a a single-layer LSTM (shown in Fig. 2) which is followed by a sigmoid layer to produce the output probabilities:

$$[h_n; c_n] = \text{LSTM}_{\text{inf}}([x_n; h_z], [h_{n-1}; c_{n-1}])$$

$$y_n = \boldsymbol{\sigma}(\boldsymbol{W}_h h_n + \boldsymbol{b}_h)$$

Since there are only two possible output values in a generated sequence {0,1}, and the sequence ends once we predict 1, the inference network can be considered an autoregressive model where 0 is passed implicitly to the subsequent time-step. To generate an output sequence, we can sample from the distribution $p(y_n = 1 | y_{R_{\text{START}}} = 0, y_{R_{\text{START}}+1} = 0, ..., y_{n-1} = 0, X_{0:n}, h_z)$ using a Bernoulli random trial at each time-step. For frames prior to R_{START} the output probability is fixed to 0, since R_{START} is the point where the DM has formulated the response. During training we minimize the binary cross entropy loss (L_{BCE}) between our ground truth objective and our output predictions Y.

2.2 RTNet-VAE

Motivation A limitation of RTNet is that it may be impractical to encode system turns before triggering a response. For example, if we wish to apply RTNet using generated system responses, at run-time the RTNet component would have to wait for the full response to be generated by the NLG, which would result in a computational bottleneck. If the NLG system is incremental, it may also be desirable for the system to start speaking before the entirety of the system response has been generated.

VAE To address this, we bypass the encoding stage by directly using the semantic representation output from the DM to control the response timing encodings. We do this by replacing the reduction layer with a VAE (Fig. 4). To train the VAE, we use the same concatenation of encoder hidden states as in the RTNet reduction layer ($[h_0; h_1; h_I]$). We use a dimensionality reduction RELU layer to calculate h_{reduce}, which is then split into μ and $\hat{\sigma}$ components via two more RELU layers. $\hat{\sigma}$ is passed through an exponential function to produce σ, a non-negative standard deviation parameter. We sample the latent variable z with the standard VAE method using μ, σ, and a random vector from the standard normal distribution $\mathcal{N}(0, \mathbf{I})$. A dimensionality expansion RELU layer is used to transform z into the response encoding h_z, which is the same dimensionality as the output of the encoder:

$$h_{\text{reduce}} = \text{RELU}(W_{\text{reduce}}[h_0; h_1; h_I] + b_{\text{reduce}})$$

$$\mu = \text{RELU}(W_\mu h_{\text{reduce}} + b_\mu)$$

$$\hat{\sigma} = \text{RELU}(W_\sigma h_{\text{reduce}} + b_\sigma)$$

$$\sigma = \exp(\frac{\hat{\sigma}}{2})$$

$$z = \mu + \sigma \odot \mathcal{N}(0, \mathbf{I})$$

$$h_z = \text{RELU}(W_{\text{expand}} z + b_{\text{expand}})$$

We impose a Gaussian prior over the latent space using a Kullback-Liebler (KL) divergence loss term:

$$L_{\text{KL}} = -\frac{1}{2N_z}(1 + \hat{\sigma} - \mu^2 - \exp(\hat{\sigma}))$$

The L_{KL} loss measures the distance of the generated distribution from a Gaussian with zero mean

and unit variance. L_{KL} is combined with L_{BCE} using a weighted sum:

$$L = L_{\text{BCE}} + w_{\text{KL}} L_{\text{KL}}$$

As we increase the value of w_{KL} we increasingly enforce the Gaussian prior on the latent space. In doing so our aim is to learn a smooth latent space in which similar types of responses are organized in similar areas of the space.

Latent Space During inference we can skip the encoding stage of RTNet-VAE and sample z directly from the latent space on the basis of the input semantic representation from the dialogue manager. Our sampling approach is to approximate the distribution of latent variables for a given response-type using Gaussians. For example, if we have a collection of labelled *backchannel* responses (and their corresponding z encodings) we can approximate the distribution of $p(z|\text{label} = backchannel)$ using an isotropic Gaussian by simply calculating $\mu_{backchannel}$ and $\sigma_{backchannel}$, the maximum likelihood mean and standard deviations of each of the z dimensions. These vectors can also be used to calculate directions in the latent space with different semantic characteristics and then interpolate between them.

2.3 Input Feature Representations

Linguistic Features We use the word annotations from the ms-state transcriptions as linguistic features. These annotations give us the timing for the starts and ends of all words in the corpus. As our feature representation, we use 300 dimensional word embeddings that are initialized with GloVe vectors (Pennington et al., 2014) and then jointly optimized with the rest of the network. In total there are 30080 unique words in the annotations. We reduced the embedding number down to 10000 by merging embeddings that had low word counts with the closest neighbouring embedding (calculated using cosine distance).

We also introduce four additional tokens that are specific to our task: SIL, WAIT, NONE, and UNSPEC. SIL is used whenever there is a silence. WAIT and NONE are used at the start and end of all the system encodings, respectively. The use of UNSPEC (unspecified) is shown in Fig. 5. UNSPEC was introduced to represent temporal information in the linguistic embeddings. We approximate the processing delay in ASR by delaying the annotation by 100 ms after the ground truth frame where the user's word ended. This 100 ms delay was proposed in Skantze (2017) as a necessary assumption to modelling linguistic features in offline continuous systems. However, since voice activity detection (VAD) can supply an estimate of when a word has started, we propose that we can use this information to supply the network with the UNSPEC embedding 100ms after the word has started.

Acoustic Features We combine 40 log-mel filterbanks, and 17 features from the GeMAPs feature set (Eyben et al., 2016). The GeMAPs features are the complete set excluding the MFCCs (e.g. pitch, intensity, spectral flux, jitter, etc.). Acoustic features were extracted using a 50ms framestep.

2.4 Experimental Settings

Training and Testing Procedures The training, validation, and test sets consist of 1646, 150, 642 conversations respectively with 151595, 13910, and 58783 turn pairs. The test set includes all of the conversations from the NXT-format annotations (Calhoun et al., 2010), which include references to the Switchboard Dialog Act Corpus (SWDA) (Stolcke et al., 2000) annotations. We include the entirety of the NXT annotations in our test set so that we have enough labelled dialogue act samples to analyse the distributions.

We used the following hyperparameter settings in our experiments: The inference, acoustic, linguistic, and master LSTMs each had hidden sizes of 1024, 256, 256, and 512 (respectively). We used a latent variable size of 4, a batch size of 128, and L2 regularization of 1e-05. We used the Adam optimizer with an initial learning rate of 5e-04. We trained each model for 15000 iterations, with learning rate reductions by a factor of 0.1 after 9000, 11000, 13000, and 14000 iterations.

While we found that randomizing R_{START} during training was important for the reasons given in Section 2, it presented issues for the stability and reproducibility of our evaluation and test results for L_{BCE} and L_{KL}. We therefore randomize during training and sampling, but when calculating the test losses (reported in Table 1) we fix R_{START} to be the first frame of the user's turn-final IPU.

We also calculate the mean absolute error (MAE), given in seconds, from the ground truth response offsets to the generated output offsets. When sampling for the calculation of MAE, it is necessary to increase the length of the turn pair since the response time may be triggered by the

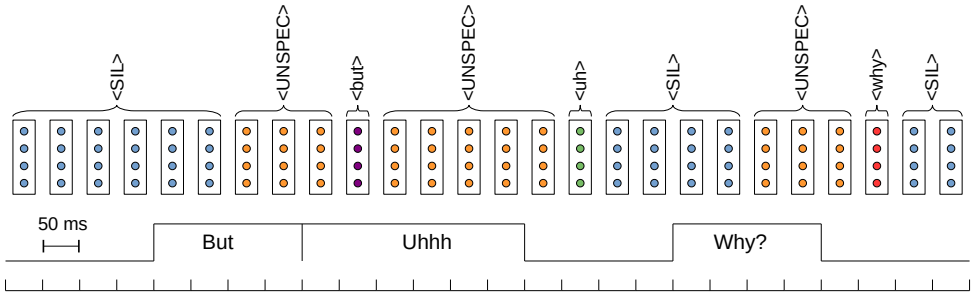
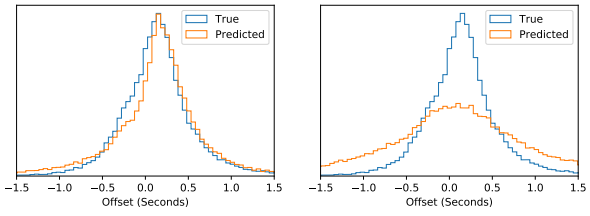


Figure 5: The user's linguistic feature representation scheme. The embedding for each word is triggered 100 ms after the ground truth end of the word, to simulate ASR delay. The UNSPEC embedding begins 100ms after a word's start frame and holds information about whether a word is being spoken (before it has been recognized) and the length of each word.

sampling process *after* the ground truth time. We therefore pad the user's features with 80 extra frames in which we simulate silence artificially using acoustic features. During sampling, we use the same R_{START} randomization process that was used during training, rather than fixing it to the start of the user's turn-final IPU. For each model we perform the sampling procedure on the test set three times and report the mean error in Table 1.

Best Fixed Probability To the best of our knowledge, there aren't any other published models that we can directly compare ours to. However, we can calculate the best performance that can be achieved using a fixed value for y. The best possible fixed y for a given turn pair is: $y_{\text{tp}} = \frac{1}{(R_{\text{END}} - R_{\text{START}})/\text{FrameLength}}$. The best fixed y for a set of turn pairs is given by the expected value of y_{tp} in that set: $y_{\text{fixed}} = \mathbb{E}[y_{\text{tp}}]$. This represents the best performance that we could achieve if we did not have access to any user or system features. We can use the fixed probability model to put the performance of the rest of our models into context.

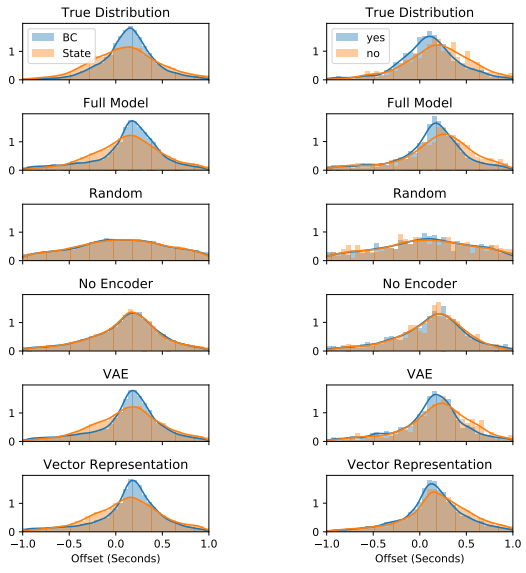


(a) Full Model (b) Fixed Probability

Figure 6: Generated offset distributions for the test set using the full model and the fixed probability (random) model.

#	Model	L_{BCE}	L_{KL}	MAE	Details
1	Full Model	**0.1094**	–	**0.4539**	No VAE
2	Fixed Probability	0.1295	–	1.4546	Fixed Probability
3	No Encoder	0.1183	–	0.4934	Encoder Ablation
4	Only Acoustic	0.1114	–	0.4627	
5	Only Linguistic	0.1144	–	0.4817	
6	Only Acoustic	0.1112	–	0.5053	Inference Ablation
7	Only Linguistic	0.1167	–	0.4923	
8	$w_{KL} = 0.0$	0.1114	3.3879	0.4601	Inclusion of VAE
9	$w_{KL} = 10^{-4}$	0.1122	1.5057	0.4689	
10	$w_{KL} = 10^{-3}$	0.1125	0.8015	0.4697	
11	$w_{KL} = 10^{-2}$	0.1181	0.0000	0.5035	
12	$w_{KL} = 10^{-1}$	0.1189	**0.0000**	0.5052	

Table 1: Experimental results on our test set. Lower is better in all cases. Best results shown in bold.



(a) BC/Statement (b) Yes/No

Figure 7: Generated offset distributions for selected response dialogue acts using different model conditions.

3 Discussion

3.1 RTNet Discussion

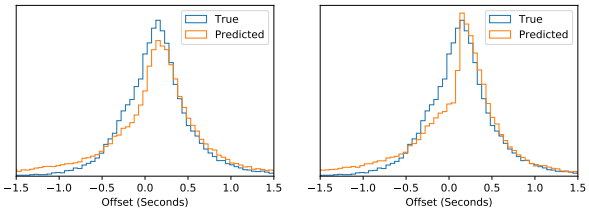
RTNet Performance The offset distribution for the full RTNet model is shown in Fig. 6a. This

baseline RTNet model is better able to replicate many of the features of the true distribution in comparison with predicted offsets using the best possible fixed probability shown in Fig. 6b. The differences between the baseline and the fixed probability distributions are reflected in the results of rows 1 and 2 in Table 1. In Fig. 6a, the model has the most trouble reproducing the distribution of offsets between -500 ms and 0 ms. This part of the distribution is the most demanding because it requires that the model anticipate the user's turn-ending. From the plots it is clear that our model is able to do that to a large degree. We observe that after the user has stopped speaking (from 0 seconds onward) the generated distribution follows the true distribution closely.

To look in more detail at how the system models the offset distribution we can investigate the generated distributions of labelled response dialogue acts in our test set. Fig. 7 shows plots of *backchannels* vs. *statements* (Fig. 7a), and *yes* vs. *no* (Fig.7b) responses. In the second rows, we can see that the full model is able to accurately capture the differences in the contours of the true distributions. For example, in the *no* dialogue acts, the full model accurately generates a mode that is delayed (relative to *yes* dialogue acts).

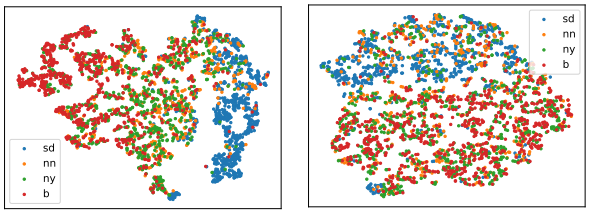
Encoder Ablation The performance of the response encoder was analysed in an ablation study, with results in rows 3 through 5 of Table 1. Without the response encoder, there is a large decrease in performance, relative to the full model. From looking at the encoders with only acoustic and linguistic modalities, we can see that the results benefit more from the acoustic modality than the linguistic modality. If we consider the impact of the encoder in more detail, we would expect that the network would not be able to model distributional differences between different types of DA responses without an encoder. This is confirmed in the fourth rows of Fig. 7, where we show the generated distributions without the encoder. We can see that without the encoder, the distributions of the all of the dialogue act offsets are almost exactly the same.

Inference Network Ablation In rows 6 and 7 of Table 1 we present an ablation of the inference network. We can see that removing either the acoustic or linguistic features from the user's features is detrimental to the results. An interesting irregularity is observed in the results for the model that uses only acoustic features (row 6): the MAE is unusually high, relative to the L_{BCE}. In all other rows, lower L_{BCE} corresponds to lower MAE. However, row 6 has the second lowest L_{BCE}, while also having the second highest MAE.



(a) Only Acoustic (b) Only Linguistic

Figure 8: Generated offset distributions for the inference network ablation.



(a) $w_{\text{KL}} = 0.0$ (b) $w_{\text{KL}} = 10^{-3}$

Figure 9: T-SNE plots of z for four different dialogue acts using two different w_{KL} settings.

In order to examine this irregularity in more detail, we look at the generated distributions from the inference ablation, shown in Fig. 8. We observe that the linguistic features are better for predicting the mode of the distribution whereas the acoustic features are better at modelling the -100 ms to +150 ms region directly preceding the mode. Since word embeddings are triggered 100 ms after the end of the word, the linguistic features can be used to generate modal offsets in the 150 ms to 200 ms bin. We propose that, in the absence of linguistic features, there is more uncertainty about when the user's turn-end has occurred. Since the majority of all ground-truth offsets occur after the user has finished speaking, the unusually high MAE in row 6 could be attributed to this uncertainty in whether the user has finished speaking.

3.2 RTNet-VAE Discussion

RTNet-VAE Performance In rows 8 through 12 of Table 1 we show the results of our experiments with RTNet-VAE with different settings of w_{KL}. As w_{KL} is increased, the L_{BCE} loss increases while the L_{KL} loss decreases. Examining some example distributions of dialogue acts generated by RTNet-VAE using $w_{\text{KL}} = 10^{-4}$ (shown in the fifth rows of Fig. 7) we can see that RTNet-VAE is capa-

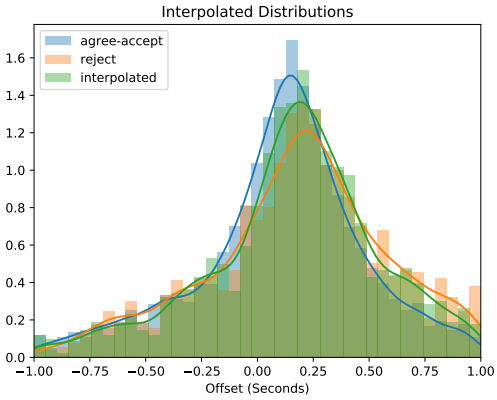


Figure 10: Interpolated distributions

ble of generating distributions that are of a similar quality to those generated by RTNet (shown in the second row). We also observe that RTNet-VAE using $w_{\mathrm{KL}} = 10^{-4}$ produces competitive results, in comparison to the full model. These observations suggest that the inclusion of the VAE in pipeline does not severely impact the overall performance.

In Fig. 9 we show the latent variable z generated using RTNet-VAE and plotted using t-SNE (van der Maaten and Hinton, 2008). To show the benefits of imposing the Gaussian prior, we show plots for with $w_{\mathrm{KL}} = 0.0$ and $w_{\mathrm{KL}} = 10^{-3}$. The plots show the two-dimensional projection of four different types of dialogue act responses: *statements* (sd), *no* (nn), *yes* (ny), and *backchannels* (b). We can observe that for both settings, the latent space is able to organize the responses by dialogue act type, even though it is never explicitly trained on dialogue act labels. For example, in both cases, statements (shown in blue) are clustered at the opposite side of the distribution from backchannels (shown in red). However, in the case of $w_{\mathrm{KL}} = 0.0$ there are "holes" in the latent space. For practical applications such as interpolation of vector representations of dialogue acts (discussed in the next paragraph), we would like a space that does not contain any of these holes since they are less likely to have semantically meaningful interpretations. When the Gaussian prior is enforced (Fig. 9b) we can see that the space is smooth and the distinctions between dialogue acts is still maintained.

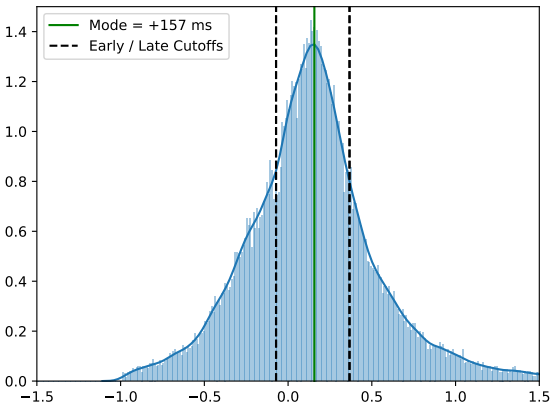
Latent Space Applications As mentioned in Section 2.2, part of the appeal in using the VAE in our model is that it enables us to discard the response encoding stage. We can exploit the smoothness of the latent space to skip the encoding stage by sampling directly from the trained latent space. We can approximate the distribution of latent variables for individual dialogue act response types using isotropic Gaussians. This enables us to efficiently represent the dialogue acts using mean and standard-deviation vectors, a pair for each dialogue act. Fig. 7 shows examples of distributions generated using Gaussian approximations of the latent space distributions in the final rows. We can see that the generated outputs have similar properties to the true distributions.

We can use the same parameterized vector representations to interpolate between different dialogue act parameters to achieve intermediate distributions. This dimensional approach is flexible in that we give the dialogue manager (DM) more control over the details of the distribution. For example, if the objective of the SDS was to generate an *agree* dialogue act, we could control the *degree* of agreement by interpolating between *disagree* and *agree* vectors. Figure 10 shows an example of a generated interpolated distribution. We can see that the properties of the interpolated distribution (e.g. mode, kurtosis) are perceptually "in between" the *reject* and *accept* distributions.

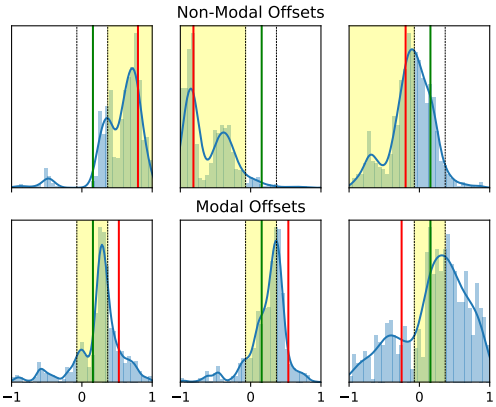
4 Listening Tests

It has shown that response timings vary based on the semantic content of dialogue responses and the preceding turn (Levinson and Torreira, 2015), and that listeners are sensitive to these fluctuations in timing (Bögels and Levinson, 2017). However, the question of whether certain response timings within different contexts are considered more *realistic* than others has not been fully investigated. We design an online listening test to answer two questions: (1) Given a preceding turn and a response, are some response timings considered by listeners to be more realistic than others? (2) In cases where listeners are sensitive to the response timing, is our model more likely to generate responses that are considered realistic than a system that generates a modal response time?

Participants were asked to make A/B choices between two versions of a turn pair, where each version had a different response offset. Participants were asked: "Which response timing sounds like it was produced in the real conversation?" The turn pairs were drawn from our dataset and were limited to pairs where the response was either *dispreferred* or a *backchannel*. We limited the chosen pairs to those with ground truth offsets that were either clas-



(a) *Early*, *modal*, and *late* regions.



(b) Generated distributions for six turn pairs. The highlighted regions indicate the region that was preferred by listeners. The red line indicates the ground truth offset.

Figure 11: Listening test experiments

sified as *early* or *late*. We classified offsets as early, modal, or late by segmenting the distribution of all of the offsets in our dataset into three partitions as shown in Fig. 11a. The cutoff points for the early and late offsets were estimated using a heuristic where we split the offsets in our dataset into two groups at the mode of the distribution (157 ms) and then used the median values of the upper (+367 ms) and lower (-72 ms) groups as the cutoff points. We selected eight examples of each dialogue act (four *early* and four *late*). We generated three different versions of each turn pair: *true*, *modal*, and *opposite*. If the true offset was late, the opposite offset was the mean of the early offsets (-316 ms). If the true offset was early, the opposite offset was the mean of the late offsets (+760 ms).

We had 25 participants (15 female, 10 male) who all wore headphones. We performed binomial tests for the significance of a given choice in each question. For the questions in the first half of the test, in which we compared *true* vs. *opposite* offsets, 10 of the 16 comparisons were found to be statistically significant ($p < 0.05$). In all of the significant cases the *true* offset was was considered more realistic than the *opposite*. In reference to our first research question, this result supports the conclusion that some responses are indeed considered to be more realistic than others. For the questions in the second half of the test, in which we compared *true* vs. *modal* offsets, six out of the 16 comparisons were found to be statistically significant. Of the six significant preferences, three were a preference for the *true* offset, and three were a preference for the *modal* offset. To investigate our second research question, we looked at the offset distributions generated by our model for each of the six significant preferences, shown in Fig. 11b. For the turn pairs where listeners preferred non-modal offsets (top row), the distributions generated by our system deviate from the mode into the preferred area (highlighted in yellow). In pairs where listeners preferred modal offsets (bottom row) the generated distributions tend to have a mode near the overall dataset mode (shown in the green line). We can conclude, in reference to our second question, that in instances where listeners are sensitive to response timings it is likely that our system will generate response timings that are more realistic than a system that simply generates the mode of the dataset.

5 Conclusion

In this paper, we have presented models that can be used to generate the turn switch offset distributions of SDS system responses. It has been shown in prior studies (e.g. (Bögels et al., 2019)) that humans are sensitive to these timings and that they can impact how responses are perceived by a listener. We would argue that they are an important element of producing naturalistic interactions that is often overlooked. With the advent of commercial SDS systems that attempt to engage users over extended multi-turn interactions (e.g. (Zhou et al., 2018)) generating realistic response behaviors is a potentially desirable addition to the overall experience.

Acknowledgments

The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

References

Sara Bögels, Kobin H. Kendrick, and Stephen C. Levinson. 2019. Conversational expectations get revised as response latencies unfold. *Language, Cognition and Neuroscience*, pages 1–14.

Sara Bögels and Stephen C. Levinson. 2017. The Brain Behind the Response: Insights Into Turn-taking in Conversation From Neuroimaging. *Research on Language and Social Interaction*, 50(1):71–89.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Sasha Calhoun, Jean Carletta, Jason M. Brenier, Neil Mayo, Dan Jurafsky, Mark Steedman, and David Beaver. 2010. The NXT-format Switchboard Corpus: A rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. *Language Resources and Evaluation*, 44(4):387–419.

Nina Dethlefs, Helen Hastie, Verena Rieser, and Oliver Lemon. 2012. Optimising incremental dialogue decisions using information density for interactive systems. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 82–93. Association for Computational Linguistics.

David DeVault, Kenji Sagae, and David Traum. 2011. Incremental interpretation and prediction of utterance meaning for interactive dialogue. *Dialogue & Discourse*, 2(1):143–170.

Florian Eyben, Klaus R. Scherer, Bjorn W. Schuller, Johan Sundberg, Elisabeth Andre, Carlos Busso, Laurence Y. Devillers, Julien Epps, Petri Laukka, Shrikanth S. Narayanan, and Khiet P. Truong. 2016. The Geneva Minimalistic Acoustic Parameter Set (GeMAPS) for Voice Research and Affective Computing. *IEEE Transactions on Affective Computing*, 7(2):190–202.

John J Godfrey and Edward Holliman. 1997. Switchboard-1 release 2. *Linguistic Data Consortium, Philadelphia*, 926:927.

David Ha and Douglas Eck. 2018. A neural representation of sketch drawings. In *International Conference on Learning Representations*.

Peter A. Heeman and Rebecca Lunsford. 2017. Turn-taking offsets and dialogue context. In *Proc. Interspeech 2017*, pages 1671–1675.

Kobin H. Kendrick and Francisco Torreira. 2015. The timing and construction of preference: A quantitative study. *Discourse Processes*, 52(4):255–289.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations*.

Divesh Lala, Pierrick Milhorat, Koji Inoue, Masanari Ishida, Katsuya Takanashi, and Tatsuya Kawahara. 2017. Attentive listening system with backchanneling, response generation and flexible turn-taking. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 127–136, Saarbrücken, Germany. Association for Computational Linguistics.

Divesh Lala, Shizuka Nakamura, and Tatsuya Kawahara. 2019. Analysis of Effect and Timing of Fillers in Natural Turn-Taking. In *Interspeech 2019*, pages 4175–4179. ISCA.

Stephen C. Levinson and Francisco Torreira. 2015. Timing in turn-taking and its implications for processing models of language. *Frontiers in Psychology*, 6.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605.

Raveesh Meena, Gabriel Skantze, and Joakim Gustafson. 2014. Data-driven models for timing feedback responses in a Map Task dialogue system. *Computer Speech & Language*, 28(4):903–922.

Louis-Philippe Morency, Iwan de Kok, and Jonathan Gratch. 2010. A probabilistic multimodal approach for predicting listener backchannels. *Autonomous Agents and Multi-Agent Systems*, 20(1):70–84.

Ryosuke Nakanishi, Koji Inoue, Shizuka Nakamura, Katsuya Takanashi, and Tatsuya Kawahara. 2018. Generating Fillers based on Dialog Act Pairs for Smooth Turn-Taking by Humanoid Robot. *IWSDS*, page 11.

Jeffrey Pennington, Richard Socher, and Christopher 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, Doha, Qatar. Association for Computational Linguistics.

Antoine Raux and Maxine Eskenazi. 2009. A finite-state turn-taking model for spoken dialog systems. In *HLT-NAACL*, pages 629–637. ACL.

Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. In *ICML*, pages 4361–4370.

Matthew Roddy, Gabriel Skantze, and Naomi Harte. 2018. Multimodal Continuous Turn-Taking Prediction Using Multiscale RNNs. In *Proceedings of the 2018 on International Conference on Multimodal Interaction - ICMI '18*, pages 186–190, Boulder, CO, USA. ACM Press.

Harvey Sacks, Emanuel A. Schegloff, and Gail Jefferson. 1974. A Simplest Systematics for the Organization of Turn-Taking for Conversation. *Language*, 50(4):696.

David Schlangen and Gabriel Skantze. 2011. A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 710–718.

Gabriel Skantze. 2017. Towards a General, Continuous Model of Turn-taking in Spoken Dialogue using LSTM Recurrent Neural Networks. In *Proceedings of SigDial*, Saarbrucken, Germany.

RJ Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron J Weiss, Rob Clark, and Rif A Saurous. 2018. Towards End-to-End Prosody Transfer for Expressive Speech Synthesis with Tacotron. *Proceedings of the 35 th International Conference on Machine Learning*, page 10.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–664, Vancouver, Canada. Association for Computational Linguistics.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2018. The Design and Implementation of XiaoIce, an Empathetic Social Chatbot. *arXiv preprint arXiv:1812.08989*.