

# Improving Dialogue Smoothing with A-priori State Pruning

Manex Serras<sup>1</sup><sup>a</sup>, María Inés Torres<sup>2</sup> and Arantza del Pozo<sup>1</sup>

<sup>1</sup>Speech and Natural Language Technologies, Vicomtech, Paseo Mikeletegi 57, Donostia-San Sebastian, Spain

<sup>2</sup>Speech Interactive Research Group, Universidad del País Vasco UPV/EHU, Campus of Leioa, Leioa, Spain

**Keywords:** Dialogue State Pruning, Dialogue Breakdown, Attributed Probabilistic Finite State Bi-Automata, Dialogue Systems.

**Abstract:** When Dialogue Systems (DS) face real usage, a challenge to solve is managing unforeseen situations without breaking the coherence of the dialogue. One way to achieve this is by redirecting the interaction to known dialogue states in a transparent way. This work proposes a simple a-priori pruning method to rule out invalid candidates when searching for similar dialogue states in unexpected scenarios. The proposed method is evaluated on a User Model (UM) based on Attributed Probabilistic Finite State Bi-Automata (A-PFSBA), trained on the Dialogue State Tracking Challenge 2 (DSTC2) corpus. Results show that the proposed technique improves response times and achieves higher F1 scores than previous A-PFSBA implementations and deep learning models.

## 1 INTRODUCTION

During the last years, technological advances in telecommunications have brought and normalized new digital communication channels. This along with the advances in Natural Language Processing, Speech Recognition, Decision Making and other Artificial Intelligence fields enabled the creation of Dialogue Systems (DS), more commonly known as Voice Assistants or Chatbots.

Dialogue Systems usually employ voice or text for engaging conversations with users whether the objective is to fulfill a task or entertain (Chen et al., 2017). As DS can automatize low-level tasks in a distributed way and allow a natural and frictionless channel to communicate with the users, it is not surprising that they have been used in multiple tasks such as bus schedule information systems (Raux et al., 2005), post-sales management (Serras et al., 2017a), coaching for elderly people (Montenegro et al., 2019; Torres et al., 2019), general entertainment (Curry et al., 2018) etc.

Every DS has to understand what it's being told, plan a coherent response strategy to fulfill the task and give an adequate response. The Dialogue Manager (DM) is the module that plans the interaction strategy according to the context and the tasks to complete. The dialogue planning can be treated as a

pattern matching, classification problem or even as a generative process, due to this, the technological stack available to build DMs employs handcrafted rules or automata (Cole, 1999; Scheffler and Young, 2000), Support Vector Machines (SVM) (Griol et al., 2008) stochastic discrete probabilistic models such as Markov Decision Process (MDP) (Zhao, 2016; Esghhi et al., 2017) and Partially Observable MDPs, Attributed Probabilistic Finite State Bi-Automata (A-PFSBA) (Torres, 2013; Serras et al., 2017b), Bayesian Networks (Pietquin and Dutoit, 2006) and more recently, both discriminative Sequence-to-Sequence Neural Networks and Generative Adversarial Networks (Agarwal et al., 2018; Lipton et al., 2018; López Zorrilla et al., 2019).

Once the DS is deployed and faces new interactions, it is common to encounter circumstances where the dialogue is led to unexpected scenarios. In these situations, the main objective of every DM is to give an appropriate response that does not induce to a dialogue breakdown (Higashinaka et al., 2016; Bohus and Rudnicky, 2005).

In the circumstance of a dialogue breakdown, it is common for the DM to employ error handling techniques such as pre-defined fallback actions (e.g. "Sorry, could you rephrase that?") (Milhorat et al., 2019; Paek and Pieraccini, 2008) or error recovery strategies that narrows down the user responses to expected options (e.g. "Select the Bus route A or B")

<sup>a</sup> <https://orcid.org/0000-0000-0000-0000>

(Bohus and Rudnicky, 2005). When it comes to Deep Learning (DL) approaches, every dialogue sequence is approximated by the networks' internal weights so the model is able to give response to any situation, being this correct or not. To detect and avoid these possible breakdowns, the Dialogue Breakdown Detection challenge has been created (Higashinaka et al., 2017), where the breakdown detection –i.e. when the system is giving an inappropriate response– is treated as a prediction problem where different algorithms such as Bi-LSTMs (Xie and Ling, 2017) and SVMs (Lopes, 2017) were used.

To avoid dialogue breakdown when an unseen interaction is reached, the A-PFSBA framework uses a dialogue smoothing strategy (Orozko and Torres, 2015; Serras et al., 2017b; Serras. et al., 2019). This strategy consists on a two step procedure where first the most similar known dialogue states to the unknown dialogue state are sampled and then, the actions of these known states are used to keep on with the dialogue, trying to transparently avoid any potential breakdown.

Unfortunately, when performing a spatial approximation of an unknown dialogue state it is imperative to correctly sample the most similar states so the given response keeps being coherent, as an incoherent response could be worse in terms of dialogue breakdown than a fallback action. Focusing on this aspect of the problem, this paper presents a simple-yet-effective a-priori state pruning methodology to improve the generalization of stochastic dialogue system when facing unknown situations. The proposed method is both applicable for DS and User Models (UM), that are Dialogue Systems that emulate real users, and are often used for data augmentation and automated testing (Schatzmann et al., 2006). Even they differ in name, the technological stack that they require is identical with the difference that the UMs have to model the final user behavior. In both scenarios the DM is a critical component to ensure the coherence of the interaction and, thus, the same dialogue state pruning method can be applied.

The proposed methodology is tested over the Dialogue State Tracking Challenge 2 (DSTC2) corpus using the A-PFSBA framework. The achieved results are compared using previous A-PFSBA models as baseline and other DL UM as a hard baseline.

The paper is structured as follows: Section 2 briefly introduces the A-PFSBA framework, how the dialogues are modelled and the dialogue smoothing strategy; Section 3 presents the dialogue a-priori state pruning method; Section 4 describes the experimental framework and the evaluations performed over the DSTC2 corpus; finally, Section 5 summarizes the

conclusions of the paper with a brief discussion and sets the guidelines for future work.

## 2 DIALOGUE MODELLING WITH ATTRIBUTED PROBABILISTIC FINITE STATE Bi-Automata

The Attributed Probabilistic Finite State Bi-Automata framework (Torres, 2013) models the interaction between an user and an agent as a composition of two alphabets, the alphabet of user actions  $d \in \Sigma$  and system actions  $a \in \Delta$ . This bi-language  $\Gamma = \Delta \times \Sigma$  is enhanced with the attribute alphabet  $\Omega$ , which encodes the transitive information of the dialogue –i.e the dialogue memory or the pieces of information that should be kept turn by turn– which is inferred directly from the dialogue interaction. Then each combination of the  $\Gamma$  and  $\Omega$  alphabets render a dialogue state  $q \in Q$  of the A-PFSBA model, which are connected by the set of transition actions  $\delta$  of the form  $(d_i : \epsilon)$  for user actions and  $(\epsilon : a_j)$  for the system actions. Each transition is fully determined by the initial and final state. As the bi-language models both user and system actions, the state set is composed by user and system states  $Q = Q_S \cup Q_U$ , depending on who has to answer. Note that all the dialogues start from the empty state  $q_0 \in Q$  which is composed by  $\epsilon$ , the empty symbol.

The A-PFSBA model is built from a sample of dialogues  $z \in Z$ , where the principal objective is to maximize the probability of the model  $\hat{M}$  to generate those dialogue samples.

$$\hat{M} = \arg \max_M P_M(Z) = \arg \max_M \prod_{z \in Z} P_M(z)$$

One of the key aspects of the A-PFSBA framework is that this formulation separates the structural learning of the dialogue samples  $Z$  and its exploitation for dialogue management. Once the model graph structure is learned, this is used to track the dialogue states through an interaction, using a policy function  $\Pi$  to select the response to be given by the DM according to the available transition edges (Ghigi and Torres, 2015; Serras et al., 2019b).

### 2.1 Dialogue Smoothing Strategy

When using the A-PFSBA model  $\hat{M}$  as DM, variances on the dialogue interactions may lead to an unknown dialogue state  $q' \notin Q$ . In this situation the policy  $\Pi$  cannot select a transition edge, as  $q'$  is not part of the model.

To keep on with the dialogue without any breakdown, a two step dialogue smoothing strategy is used: first, the most similar known dialogue states are found using a  $G$  spatial function –which can be the Euclidean distance or cosine similarity–; then, the next action is selected using these known dialogue states (Orozko and Torres, 2015; Ghigi and Torres, 2015; Serras. et al., 2019). This allows to recover from the unknown state  $q'$  transparently, keeping on with the dialogue without interruption. This strategy is illustrated at Figure 1. As the first step of the smoothing

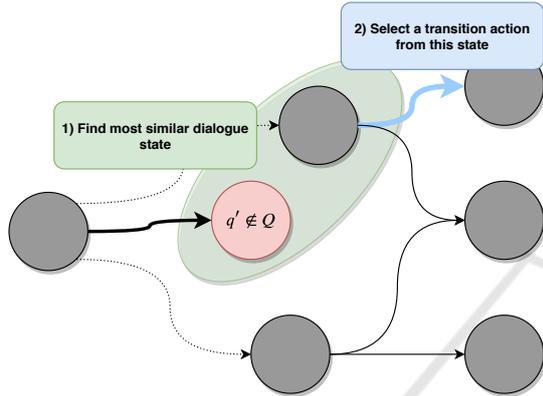


Figure 1: Dialogue smoothing procedure from the unknown dialogue state  $q'$  to select the systems' next action without breakdown.

is about sampling the nearest dialogue states using a spatial relation function  $G$ , their spatial representation is crucial. Early works on A-PFSBA employed string based state representations (Orozko and Torres, 2015; Ghigi and Torres, 2015), more recently a vectorial representation was proposed in (Serras. et al., 2019), allowing to operate in  $\mathbf{R}^n$ . Nevertheless, representing the A-PFSBA dialogue states  $q \in Q$  as vectors  $\vec{q}$  has some drawbacks, as the semantic relationships between each dimension of the vector will be ignored by common spatial relations such as Euclidean distance and cosine similarity. As an example, let us have these three actions:

1. Request(address). "Give me the address"
2. Request(telephone). "Give me the phone."
3. Inform(area=city center). "I want some restaurant at the city center"

Let us suppose that binary vector representation of these user dialogue actions are  $[1, 0, 0]$ ,  $[0, 1, 0]$  and  $[0, 0, 1]$ . In this scenario, we can observe that the Euclidean distance between these representations is the same, but their semantic meaning is completely different, and so, the response to give by a DS to each user action should be different.

Afterwards, when the first step of the dialogue

smoothing strategy is performed, the closest dialogue states are selected, but without taking into account the semantic information that is encoded in each dialogue state. This may lead to sampling spatially close but semantically unrelated dialogue states when selecting the next response, thus returning an incoherent response that causes a dialogue breakdown.

### 3 A-PRIORI STATE PRUNING

To improve the dialogue state sampling when performing the dialogue smoothing procedure, a simple a-priori dialogue state pruning method is presented in this section. The principal objective of the proposed method is to remove semantically unrelated dialogue states to the unknown state  $q'$  before the dialogue smoothing is applied. To achieve so, a function that captures the semantic relations of the vectorized dialogue-state representations  $\vec{q} \in \vec{Q}$ , is built, which is used afterwards for the dialogue-state pruning. The principal assumption of this method is that semantically similar dialogue states will return similar actions.

Generally, we can define a Pruning Model (PM) as a function that receives the vector form of a dialogue state (known or unknown)  $\vec{q}$  and gives a score for each item of the alphabet  $a \in \Delta$ .

$$PM(\vec{q}) \rightarrow \{score(a_i) \forall a_i \in \Delta\}$$

The building of this PM can be done using off-the-shelf Machine Learning (ML) algorithms. The ML model is trained with each dialogue state vector  $\vec{q}$  using its output actions  $\delta(q)$  as labels, so the model is learning the discriminative patterns of the dialogue state vector and its output actions.

Using the PM and a threshold  $\theta_{prune}$ , when some unknown state  $q'$  is reached during the conversation, the action-score distribution is predicted  $PM(\vec{q}')$ . Next, for each action  $a_j \in \Delta$ , if the score received is lower than the defined  $\theta_{prune}$  threshold, all the dialogue states that include  $a_j$  in their possible transitions  $q_i \in Q : a_j \in \delta(q_i)$  are excluded from the dialogue smoothing sampling process. Note that  $\delta(q_i)$  is the set of transitions that depart from  $q_i$  and can be used as the next system response.

As the proposed pruning method is applied to the dialogue management, it can be used by both DS and UM just by changing the dialogue state set to the user states  $Q_U$  and the action set to the user actions  $d \in \Sigma$ .

Table 1: Act/Slot level global evaluation metrics over the DSTC2 corpus for different pruning methods.

| Pruning Model          | Best $\theta_{prune}$ | Development set    |                    |                    | Test set           |                    |                    |
|------------------------|-----------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
|                        |                       | Precision          | Recall             | F1 Score           | Precision          | Recall             | F1 Score           |
| <i>baseline</i> : BAUM | –                     | 0.690/0.566        | 0.731/0.591        | 0.710/0.578        | 0.699/0.556        | 0.728/0.577        | 0.712/0.566        |
| BAUM - MNB             | 0.1                   | 0.717/0.590        | 0.746/0.605        | 0.731/0.598        | 0.747/0.594        | 0.744/0.586        | 0.746/0.590        |
| BAUM - SVM             | 0.2                   | <b>0.766/0.654</b> | <b>0.750/0.630</b> | <b>0.758/0.642</b> | <b>0.790/0.662</b> | 0.756/0.625        | <b>0.772/0.643</b> |
| BAUM - PA              | -1.6                  | 0.733/0.610        | 0.739/0.605        | 0.736/0.607        | 0.749/0.607        | 0.742/0.595        | 0.745/0.600        |
| BAUM - MLP             | 0.2                   | 0.752/0.638        | <b>0.753/0.626</b> | 0.752/0.631        | 0.774/0.638        | <b>0.761/0.618</b> | 0.768/0.628        |
| BAUM - RF              | 0.3                   | 0.757/0.630        | 0.705/0.579        | 0.730/0.603        | 0.770/0.630        | 0.703/0.570        | 0.735/0.599        |
| Reg. Bi-LSTM           | –                     | 0.70/0.60          | 0.72/ <b>0.63</b>  | 0.71/0.62          | 0.71/0.60          | 0.73/ <b>0.64</b>  | 0.72/0.62          |

Table 2: Results achieved by selecting the next action with different ML algorithms.

| <i>Predicting the Next Action</i> | Development set    |                    |                    | Test set           |                    |                    |
|-----------------------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| Prediction Algorithm              | Precision          | Recall             | F1 Score           | Precision          | Recall             | F1 Score           |
| MNB                               | 0.649/0.515        | 0.706/0.556        | 0.676/0.535        | 0.700/0.489        | 0.730/0.500        | 0.715/0.490        |
| SVM                               | 0.732/0.609        | 0.711/0.582        | 0.721/0.595        | 0.767/0.631        | 0.721/0.586        | 0.743/0.607        |
| PA                                | 0.698/0.546        | 0.689/0.527        | 0.694/0.537        | 0.725/0.531        | 0.701/0.502        | 0.713/0.516        |
| MLP                               | 0.730/0.611        | 0.732/0.604        | 0.731/0.607        | 0.770/0.613        | 0.752/0.590        | 0.760/0.60         |
| RF                                | 0.703/0.570        | 0.701/0.562        | 0.702/0.566        | 0.704/0.551        | 0.693/0.542        | 0.699/0.547        |
| BAUM - SVM                        | <b>0.766/0.654</b> | <b>0.750/0.630</b> | <b>0.758/0.642</b> | <b>0.790/0.662</b> | <b>0.756/0.625</b> | <b>0.772/0.643</b> |

## 4 EXPERIMENTATION FRAMEWORK

This section performs the experiments over the DSTC2 corpus to test the proposed dialogue state pruning method. To compare with previous results, the pruning is applied to the A-PFSBA User Model (BAUM) presented at (Serras. et al., 2019) which uses cosine similarity as spatial function  $G$  and Maximum Likelihood as policy  $\Pi$ . Several ML algorithms are used to perform the state pruning and the obtained results are compared with the results achieved by a Deep Learning Bi-LSTM ensemble UM presented at (Serras et al., 2019a). During the experimental section, three main hypotheses are tested:

1. Dialogue state pruning before the dialogue smoothing sampling improves the generalization of the Dialogue Management module.
2. The presented methodology can be applied using a wide-range of ML methods, thus the PM can be adjusted to domains that may have different constraints and features (e.g. hardware/scalability constraints, decoding time constraints, different data distributions, and so on).
3. The use of ML algorithms is better suited for state-pruning than for decision making, i.e. using the same ML algorithm to predict the next action when an unknown state  $q^i$  is reached.

### 4.1 Dialogue State Tracking Challenge 2 Corpus

The Dialogue State Tracking Challenge 2 (DSTC2) corpus was released in the second edition of the DSTC series (Henderson et al., 2014), which was focused on tracking the dialogue state of a SDS in the Cambridge restaurant domain. For such purpose, a corpus with a total of 3235 dialogues was released<sup>1</sup>. Amazon Mechanical Turk was used to recruit users who would interact with a spoken dialogue system. Previous to each dialogue, each user was given a goal to fulfill, which had to be completed interacting with the system. The goals defined in this corpus followed the agenda approach of (Schatzmann et al., 2007a; Schatzmann et al., 2007b), where the user was given some restaurant related constraints such as food type and price range and some information to request to the system, as the venue address. More information about this corpus can be found at (Henderson et al., 2013). As this corpus is already partitioned in train, development and test sets, such splits have been used in this section.

### 4.2 Impact on User Modeling

Following the evaluation procedure used by (El Asri et al., 2016; Serras et al., 2019a; Serras. et al., 2019),

<sup>1</sup><http://camdial.org/~mh521/dstc/>

the UMs are evaluated in direct comparison, i.e. comparing the output given by the UM with the real users' output. The evaluation is measured in terms of precision, recall and F1 score. The BAUM used as baseline selects just the nearest dialogue state when performing the dialogue smoothing strategy, (**BAUM**) using the cosine similarity and the maximum likelihood to select the next action. This initial baseline is enhanced with the proposed dialogue state pruning method using five well-known off-the-shelf ML algorithms. The PM is trained using the A-PFSBA model  $\tilde{M}$  inferred from the DSTC2 training set. The  $\theta_{prune}$  threshold is selected using the development set and performing a grid search with step size of 0.1 to optimize the F1 score.

The ML algorithms employed in this section are the Multinomial Naive Bayes (**MNB**) (Schütze et al., 2008), Support Vector Machines with linear kernel function (**SVM**) (Platt, 1999), Passive Aggressive Classifier (**PA**) (Crammer et al., 2006), Multi-Layer Perceptron (Hinton, 1990) (**MLP**) and Random Forest Classifier (**RF**) (Breiman, 2001). All these algorithms were implemented using scikit-learn (Pedregosa et al., 2011) and for the sake of simplicity, given that the goal is to prove the ease of use of the state pruning, no hiperparameter tuning was performed. As a hard baseline, one of the latest DL user modelling approach based on an ensemble of regularized Bi-LSTM (Serras et al., 2019a) is also included (**Reg. Bi-LSTM**).

Table 1 shows the achieved scores for each method, evaluated at intent level –higher level, coarser granularity– and slot-value level –finer granularity and more challenging–. As it is clearly seen, the inclusion of a pruning mechanism improves the baseline regardless of the ML algorithm used to build the PM. The best suited ML algorithms for the DSTC2 case happened to be the SVM and the MLP, where the results obtained with the BAUM - SVM/MLP achieved higher scores than the Deep Learning Bi-LSTM ensemble in most of the metrics. Finally, Figure 2 compares the relative decoding times achieved by the different BAUM versions in the development and test sets of the corpus. The development set decoding time of the baseline **BAUM**, which uses no PM at all, is used as reference. The results clearly indicate that using the proposed dialogue state pruning method before performing the dialogue smoothing procedure reduces the decoding time. This is a critical feature for any DS, as they have to provide a real-time service.

### 4.3 Predicting the Next Action

In the following experiment the third hypothesis presented in Section 4 is tested. As previous works on DS used ML algorithms to perform the dialogue management (Griol et al., 2008), one of the questions that arises when using one of these algorithms to prune possible dialogue states is "Why not use the ML algorithm as predictor for the next action when an unknown state  $q'$  is reached?". To justify the use of these ML methods as state pruners instead of predictors, Table 2 displays the results achieved by using the same algorithms as next user action predictors when an unknown state is reached.

As it can be seen, the overall results when using a ML algorithm to predict the next action are worse than using it for state pruning. In addition, the achieved results are less robust as the selection of the ML algorithm strongly conditions the results obtained.

## 5 CONCLUSIONS AND FUTURE WORK

In this paper a dialogue state pruning method was presented to prune semantically unrelated dialogue states when performing the dialogue smoothing procedure to avoid dialogue breakdown when an unknown dialogue state is reached during interaction. Using the DSTC2 corpus and the user modelling task for evaluation, the presented pruning method has proven to be a robust way to improve the generalization of the DM when some unknown state is reached in a dialogue interaction. In addition, the decoding time of the system is improved, mainly due because the amount of valid dialogue states is highly reduced in comparison to the original dialogue smoothing sampling method, which takes into account all the system/user dialogue states to compute the spatial similarity function  $G$ .

When using the same ML algorithms for predicting the next action –instead for state pruning– when an unknown dialogue state is met, despite some algorithms such as SVM and MLP achieve competitive results, the overall result is worse and the achieved results show a high variance depending on which algorithm is used. On the other hand, when using these algorithms for pruning, the overall score of the baseline UM is always improved. Anyways, with the presented experiments we cannot discard that the use of more complex and resource-demanding prediction algorithms and/or performing some exhaustive hyperparameter tuning on them could not match the results achieved by the PMs (note that the same effort could

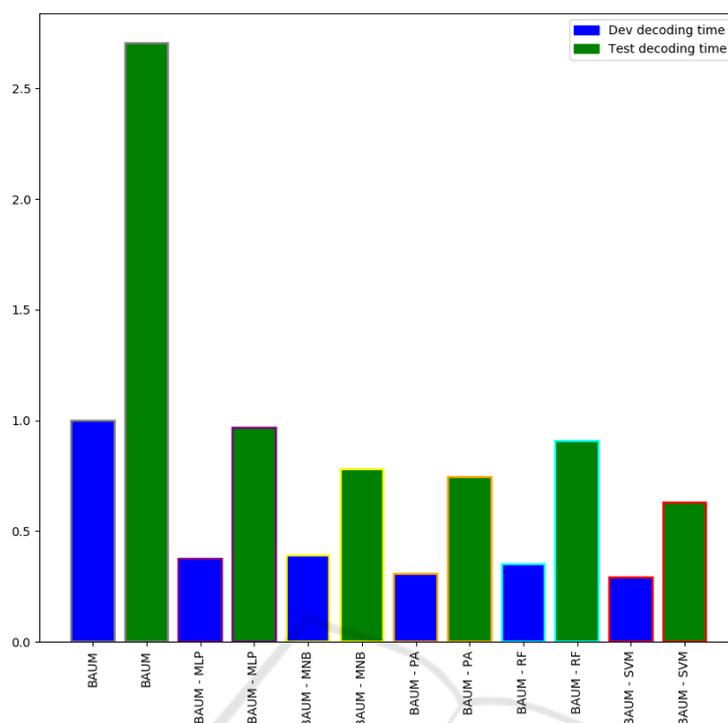


Figure 2: Relative time comparison between using different algorithms as PM in the A-PFSBA User Model.

improve the results of the dialogue state pruning), but this was not the goal of the experimental section, as one of the main points of the proposed method is its simplicity and ease of use. That is the reason why no complex algorithms were used nor hyper-parameter fine-tuning was performed.

Yet it is true that the proposed method has no direct application within the field of DL-based dialogue management, it can be used in any discrete state based DMs such as MDPs, POMDPs, HMMs and A-PFSBA. These approaches are more popular to build DS in environments where the amount of data is scarce.

Future work will be focused on performing an indirect evaluation using the proposed methodology, i.e. evaluating the Task Completion or Dialogue Success rate between dialogues generated by a UM and DM talking to each other. In addition, as dialogue state pruning reduces the number of times that the spatial function  $G$  needs to be computed when the smoothing is performed, the construction of more complex spatial relation functions –rather than using the Euclidean distance or cosine similarity– will be researched. Finally, the development of a method that decides when to select an explicit fallback action “*I don’t understand what you mean, could you rephrase that?*” instead of performing the dialogue smoothing strategy (because there is no adequate response given by the

sampled dialogue states), is an open question that would require an experimental setup with real users.

## REFERENCES

- Agarwal, S., Dušek, O., Konstas, I., and Rieser, V. (2018). Improving context modelling in multimodal dialogue generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 129–134.
- Bohus, D. and Rudnicky, A. I. (2005). Error handling in the ravenclaw dialog management framework. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 225–232. Association for Computational Linguistics.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.
- Chen, H., Liu, X., Yin, D., and Tang, J. (2017). A survey on dialogue systems: Recent advances and new frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2):25–35.
- Cole, R. (1999). Tools for research and education in speech science. In *Proceedings of the International Conference of Phonetic Sciences*, volume 1, pages 277–1. Citeseer.
- Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585.

- Curry, A. C., Papaioannou, I., Suglia, A., Agarwal, S., Shalymov, I., Xu, X., Dušek, O., Eshghi, A., Konstas, I., Rieser, V., et al. (2018). Alana v2: Entertaining and informative open-domain social dialogue using ontologies and entity linking. *Alexa Prize Proceedings*.
- El Asri, L., He, J., and Suleman, K. (2016). A sequence-to-sequence model for user simulation in spoken dialogue systems. *Interspeech 2016*, pages 1151–1155.
- Eshghi, A., Shalymov, I., and Lemon, O. (2017). Bootstrapping incremental dialogue systems from minimal data: the generalisation power of dialogue grammars. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2220–2230.
- Ghigi, F. and Torres, M. I. (2015). Decision making strategies for finite-state bi-automaton in dialog management. In *Natural Language Dialog Systems and Intelligent Assistants*, pages 209–221. Springer.
- Griol, D., Hurtado, L. F., Segarra, E., and Sanchis, E. (2008). A statistical approach to spoken dialog systems design and evaluation. *Speech Communication*, 50(8-9):666–682.
- Henderson, M., Thomson, B., and Williams, J. (2013). Dialog state tracking challenge 2 & 3.
- Henderson, M., Thomson, B., and Williams, J. D. (2014). The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272.
- Higashinaka, R., Funakoshi, K., Inaba, M., Tsunomori, Y., Takahashi, T., and Kaji, N. (2017). Overview of dialogue breakdown detection challenge 3. *Proceedings of Dialog System Technology Challenge*, 6.
- Higashinaka, R., Funakoshi, K., Kobayashi, Y., and Inaba, M. (2016). The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3146–3150.
- Hinton, G. E. (1990). Connectionist learning procedures. In *Machine learning*, pages 555–610. Elsevier.
- Lipton, Z., Li, X., Gao, J., Li, L., Ahmed, F., and Deng, L. (2018). Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Lopes, J. (2017). How generic can dialogue breakdown detection be? the kth entry to dbdc3. *Proceedings of Dialog System Technology Challenge*, 6.
- López Zorrilla, A., De Velasco Vázquez, M., and Torres Barañano, M. I. (2019). A differentiable generative adversarial network for open domain dialogue.
- Milhorat, P., Lala, D., Inoue, K., Zhao, T., Ishida, M., Takanashi, K., Nakamura, S., and Kawahara, T. (2019). A conversational dialogue manager for the humanoid robot erica. In *Advanced Social Interaction with Agents*, pages 119–131. Springer.
- Montenegro, C., Lpez Zorrilla, A., Mikel Olaso, J., Santana, R., Justo, R., Lozano, J. A., and Torres, M. I. (2019). A dialogue-act taxonomy for a virtual coach designed to improve the life of elderly. *Multimodal Technologies and Interaction*, 3(3):52.
- Orozko, O. R. and Torres, M. I. (2015). Online learning of stochastic bi-automaton to model dialogues. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 441–451. Springer.
- Paek, T. and Pieraccini, R. (2008). Automating spoken dialogue management design using machine learning: An industry perspective. *Speech communication*, 50(8-9):716–729.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- Pietquin, O. and Dutoit, T. (2006). A probabilistic framework for dialog simulation and optimal strategy learning. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(2):589–599.
- Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press.
- Raux, A., Langner, B., Bohus, D., Black, A. W., and Eskenazi, M. (2005). Let's go public! taking a spoken dialog system to the real world. In *Ninth European conference on speech communication and technology*.
- Schatzmann, J., Thomson, B., Weilhammer, K., Ye, H., and Young, S. (2007a). Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152. Association for Computational Linguistics.
- Schatzmann, J., Thomson, B., and Young, S. (2007b). Statistical user simulation with a hidden agenda. *Proc SIGDial, Antwerp*, 273282(9).
- Schatzmann, J., Weilhammer, K., Stuttle, M., and Young, S. (2006). A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The knowledge engineering review*, 21(2):97–126.
- Scheffler, K. and Young, S. (2000). Probabilistic simulation of human-machine dialogues. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 2, pages II1217–II1220. IEEE.
- Schütze, H., Manning, C. D., and Raghavan, P. (2008). Introduction to information retrieval. In *Proceedings of the international communication of association for computing machinery conference*, page 260.
- Serras, M., Perez, N., Torres, M. I., and Del Pozo, A. (2017a). Entropy-driven dialog for topic classification: detecting and tackling uncertainty. In *Dialogues with Social Robots*, pages 171–182. Springer.

- Serras, M., Torres, M. I., and Del Pozo, A. (2017b). Online learning of attributed bi-automata for dialogue management in spoken dialogue systems. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 22–31. Springer.
- Serras, M., Torres, M. I., and del Pozo, A. (2019). Goal-conditioned user modeling for dialogue systems using stochastic bi-automata. In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM*, pages 128–134. INSTICC, SciTePress.
- Serras, M., Torres, M. I., and Del Pozo, A. (2019a). Regularized neural user model for goal-oriented spoken dialogue systems. In *Advanced Social Interaction with Agents*, pages 235–245. Springer.
- Serras, M., Torres, M. I., and Del Pozo, A. (2019b). User-aware dialogue management policies over attributed bi-automata. *Pattern Analysis and Applications*, 22(4):1319–1330.
- Torres, M. I. (2013). Stochastic bi-languages to model dialogs. In *Proceedings of the 11th international conference on finite state methods and natural language processing*, pages 9–17.
- Torres, M. I., Olaso, J. M., Glackin, N., Justo, R., and Chollet, G. (2019). A spoken dialogue system for the empathic virtual coach. In *9th International Workshop on Spoken Dialogue System Technology*, pages 259–265. Springer.
- Xie, Z. and Ling, G. (2017). Dialogue breakdown detection using hierarchical bi-directional lstms. In *Proceedings of the Dialog System Technology Challenges Workshop (DSTC6)*.
- Zhao, T. (2016). Reinform: Multi-domain dialogue management using hierarchical policies and knowledge ontology.