Human Language Technologies – The Baltic Perspective
A. Utka et al. (Eds.)

© 2020 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA200607

Interactive Learning of Dialog Scenarios from Examples

Daiga DEKSNE and Raivis SKADIŅŠ [1]
Tilde, Riga, Latvia
Faculty of Computing, University of Latvia, Latvia

Abstract. This paper reports on the development of a toolkit that enables collecting dialog corpus for end-to-end goal-oriented dialog system training. The toolkit includes the neural network model that interactively learns to predict the next virtual assistant (VA) action from the conversation history. We start with exploring methods for VA dialog scenario learning from examples after we perform several experiments with the English DSTC dialog sets in order to find the optimal strategy for neural model training. The chosen algorithm is used for training the next action prediction model for the Latvian dialogs in the public transport inquiries domain collected using the platform. The accuracy for the English and the Latvian dialog models is similar – 0.84 and 0.86. This shows that the chosen method for neural network model training is language independent.

Keywords. Virtual assistants, machine learning, dialog corpus

1. Introduction

A lot of manual human effort still needs to be invested to develop virtual assistants that can help specialists in customer service. Organizations serving their clients have accumulated conversation archives in both text and audio format, yet only a small portion of that data can be used for VA training [1]. Today, VAs typically work according to dialog scenarios that are executed depending on the user's intents and the data collected from the user. VAs analyze user input, determine user intent and entities, and perform dialog scenario steps. Currently, machine learning techniques are used to train the models for intent detection and entity recognition, but dialog scenarios are usually created manually.

In this research, we explore the methods that will allow VA dialog scenarios to be learned from examples. There are several studies for English in this area, such as [2]-[8]. We aim to create the methods that are as much language-independent as possible so that they can be used for the Baltic as well as other languages.

To gather the training data, we have created an environment for interactive dialog data collection. We employ the Wizard-of-Oz approach. At first, VA users communicate with a VA trainer (person pretending to be a VA), and we log these communications. VA trainer defines new VA actions and tries to reuse already defined actions as much as possible. Then we use the collected data to build a model that predicts the next VA action. The model then is used in the data collection process to predict VA actions in new dialogs,

[1] Corresponding Author: Raivis Skadiņš; Tilde, Vienibas gatve 75a, Riga, Latvia, LV1004; E-mail: raivis.skadins@tilde.lv.

where VA trainer either picks actions predicted by the model or creates new actions. We regularly retrain the model with the new data collected, thus model quality increases while we collect new data.

2. VA Action Prediction Model

There is no open-source dialog data available for Latvian. Instead, there are several valuable dialog corpora for English:

- Cambridge Restaurant Corpus [6] was designed to assist the user to find a restaurant in the vicinity of Cambridge. The corpus includes 676 dialogs.
- Stanford Driver's In-car Assistant Dialog Dataset [9] is a multi-domain corpus in three domains: calendar scheduling, weather information retrieval, and point-of-interest navigation. The corpus contains 3,031 dialogs.
- Ubuntu Dialog Corpus [10] contains 930,000 dialogs extracted from the Ubuntu chat logs in which support for various Ubuntu-related technical problems is provided.
- JD Customer Service Corpus [11] includes online retailing customer service dialogs between customers and customer service staff on the web site JD.com. There are 415,000 dialogs for training, 1,500 dialogs for validation and 5,005 dialogs for test.
- DSTC2 and DSTC3 data sets were assembled for the Dialog State Tracking Challenge[2], where participants developed state tracking algorithms using a labeled corpus of dialogs in the restaurant information domain. In total, there are 5,506 dialogs in both data sets.
- MultiWOZ Corpus [12] contains 3,406 single domain and 7,032 multi-domain task-oriented dialogs collected through the crowd work using the Wizard of Oz method.

To achieve the goals of our research, we started by building a VA action prediction model for English. We used English data to design and validate neural network models and to experiment with different training data encodings and model hyperparameters before we start collecting data for Latvian to avoid potential mistakes in the data collection process.

2.1. Training Data

For our experiments, we used DSTC2 and DSTC3 dialog sets. We transformed the data to fit the needs of our task – prediction of the next VA action. Each dialog has dictionary type records representing a single turn in a dialog. The dialog consists of several turns. Each turn has the following information:

- **actor**: ‘VA’ or ‘user’;
- **utterance**: VA or user utterance;
- **entities**: key/value pairs representing entities and their values;

[2] http://camdial.org/~mh521/dstc/

- **intents**: an array with one or several intents (only for the user's turns);
- **action**: an action of VA (only for VA turns).

There are two types of actions that the VA can perform. One that returns a textual response, another one that performs some function potentially involving outside sources – checking availability or inquiring about something, making calculations, and setting variables.

2.2. The Architecture of the Neural Network Model

We train the neural network model that predicts the next VA action when given the previous conversation history. We use the LSTM recurrent neural network architecture for this task (see **Figure 1**). We use a single layer of 100 LSTM cells. To avoid overfitting, we introduce a Dropout Layer with a dropout rate of 0.5 after the LSTM layer. We use softmax activation in the Dense Layer, categorical cross-entropy function for loss calculation, and Adam optimizer [13].

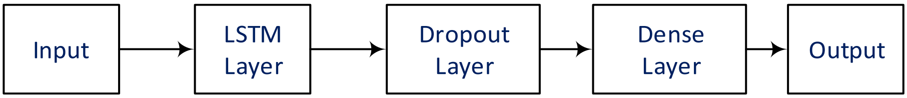


Figure 1. Layers of the neural network model

2.3. Training Settings

We performed nine experiments with different embeddings for utterance encoding and different ways how we encode the entities (see **Table 1**). The test accuracy is calculated using 10-fold cross-validation.

We evaluated two variants of representing the user input:

- One-hot vector of intents in experiments 4 and 9;
- Sentence embeddings in experiments 1-3 and 5-8.

In experiments 1, 5, and 8, we used pre-trained *fastText* embeddings [14] trained on *Wikipedia*, whereas in experiments 2, 6, and 7, we used a variation of *fastText* algorithm [15]. In experiment 3, we used uncased *BERT-Base* model [16], but as it required more computing resources and did not show better results compared to models with *fastText* embeddings, we did not use it in further experiments.

Table 1. Results of different neural network models

Nr	Vectorization	Dimensions	Entities	Test accuracy
1	wiki.en.bin	300	194	**0.8395**
2	news.wiki.en.bin	300	194	0.8303
3	BERT-Base	768	194	0.8209
4	Intents	42	194	0.8360
5	wiki.en.bin	300	20	0.8191
6	news.wiki.en.bin	300	249	0.8291
7	news.wiki.en.bin	300	498	0.8388
8	wiki.en.bin	300	498	**0.8398**
9	Intents	42	498	**0.8443**

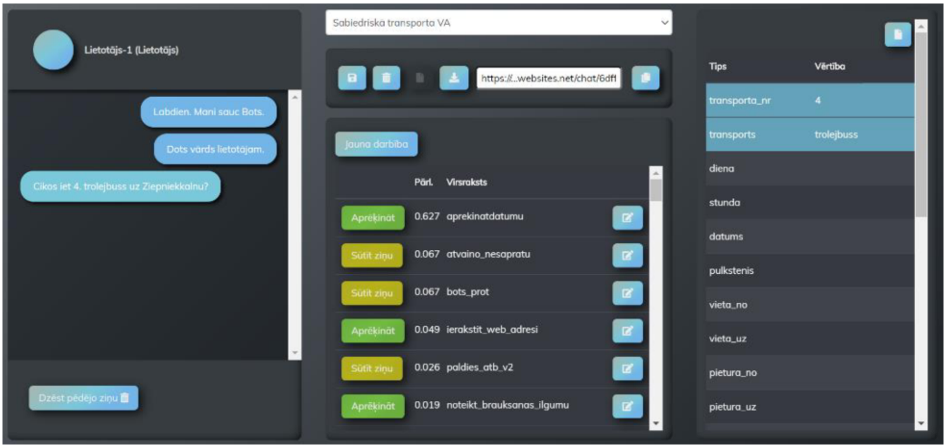
We also evaluated several variants of encoding the entities and their values:

- We made a distinction between entities and their values that are provided by the user and entities that are provided by the VA in experiments 1-4;
- We used the same set of entities for the user and the VA in experiment 6;
- We used only entity types set by the user ignoring entity values in experiment 5;
- In experiments 7-9, we used not only entities that have been set in the last turn of a dialog, but also entities set in the previous turns.

The best results are obtained by including entities from previous turns in the current turn's input, though the results of experiments 8 and 1 are not significantly different.

3. Interactive Dialog Data Collection Environment

We have designed a platform for dialog data collection. It has two parts – the VA trainer environment and the environment for user-bot communication. Both the VA trainer and the VA user client-side environment interfaces are developed using TypeScript based JavaScript framework Angular 8[3]. The server-side solution uses the .NET Core 3.1 framework[4]. Interaction between the parts of the developed platform is provided with the SignalR asynchronous Data Processing Library[5]. Dialogs collected using the platform are stored in the SQL Server database.



Labdien. Mani sauc Bots.

Dots vārds lietotājam.

Cikos iet 4. trolejbuss uz Ziepniekkalnu?

	Pārl.	Virsraksts
Aprēķināt	0.627	aprekinatdatumu
Sūtīt ziņu	0.067	atvaino_nesapratu
Sūtīt ziņu	0.067	bots_prot
Aprēķināt	0.049	ierakstit_web_adresi
Sūtīt ziņu	0.026	paldies_atb_v2
Aprēķināt	0.019	noteikt_brauksanas_ilgumu

Tips	Vērtība
transporta_nr	4
transports	trolejbuss
diena	
stunda	
datums	
pulkstenis	
vieta_no	
vieta_uz	
pietura_no	
pietura_uz	

Figure 2. VA trainer environment

The user environment has a chat window where a user can see all previous turns of a dialog and type in an utterance.

The VA trainer environment has three panels.

- On the left, turns of a current dialog are displayed.

[3] https://angular.io/

[4] https://docs.microsoft.com/en-us/dotnet/core/

[5] https://github.com/dotnet/aspnetcore/tree/master/src/SignalR

- In the middle, the VA couch can start a new dialog, save or discard a previous dialog, create new actions, and send a response to a user. At first, the action list is populated manually and all actions in the list have equal confidence index 0. After collecting an initial set of dialogs, we train a neural model for next action prediction. In further work, the model returns the list of most probable actions based on the previous conversation history. Actions are sorted by confidence index in descending order. The actions that are most appropriate for context are at the top of the list.
- On the right, the VA couch can define new entities or variables and see their values assigned in a process of conversation.

We use two types of actions. Text actions return a simple textual answer, for example, greet the user or express gratitude. Calculation actions receive variable values in the input and set one or more variable values in the output. In the future, calculation actions will be performed by external services. At this time, the values of entities and variables are set manually by the VA trainer. Examples of calculation actions are determining current time, checking the availability of some product or service, and calculating the total price of something.

4. First Experiments with Latvian Dialogs

We have started collecting dialogs for Latvian using the developed tools. Our chosen domain is transport inquiries. For now, we have collected 160 dialogs in this domain. The set of entities for the transport inquiries domain includes from/to location, transport type, route, date and time, journey length, and timetable, 23 entities in total. We have defined 29 textual actions and 14 calculation actions.

4.1. Data Collection Process

We can distinguish three steps in dialog data collection in transport inquiries domain:

- Learning what type of information is available on the public transport web page, defining types of entities and variables, and the initial set of actions to cover the domain.
- For collecting an initial set of dialogs, the VA couch played both roles – bot's and user's. In this process, the actions and entities were refined in order to improve conversation flow.
- The neural model for the next action prediction was trained and included in a workflow. Dialogs were performed between the VA trainer and different conversational partners.

Similar steps can be employed for dialog collection in any new domain.

4.2. Processing Some More Sophisticated User Utterances

In a conversation related to transport inquiries, a user may be free to ask about the timetables and routes of public transport. An "easy-to-understand" statement with all the input data required is the one containing the transport number, its type, and the stops from/to. The system does not need to ask additional questions to achieve the goal as, for example, in the statement ‘What time is the 7th tram from the Ausekļa street to the National theater?’. More problematic are the statements with incomplete information:

- missing information about the start or destination stop;
- a route that involves a change of transport for one or several times;
- instead of a stop name, the user refers to an object or place name used in colloquial speech;
- the user specifies uncertain time value that needs additional calculation, for example, ‘late in the evening’, ‘next Monday’, ‘as early as possible’.

4.3. Training Next Action Prediction Model in Transport Inquiries Domain

The next action prediction model’s architecture for dialogs in Latvian is similar to the one used in experiments with the English data. We use the *fastText* embedding module that is trained on Latvian Wikipedia texts. The maximal number of epochs is 100 with early stopping if accuracy does not improve more than by 0.0001 for three subsequent epochs. The structure of each record in training data representing each turn in a conversation is the following:

- 41 positions are allocated for the one-hot vector representing actions. Only one position in every record has value ‘1’ as there is a single action per turn. For the turns made by a user, the action is always the same meaning, i.e. ‘the user has the word’;
- 2 positions are reserved for the role of an actor – ‘user’ or VA;
- 66 positions are reserved for entity types. Both conversation partners can set none to several entities in their turns. Entity values are not used for training;
- 66 positions are reserved for entity types set in all previous dialog turns;
- 300 positions take the embedding vector. If a turn is made by a user it is the user’s utterance’s embedding vector. If it is VA’s turn, all positions have value ‘0’.

Table 2. Results of 10-fold cross-validation

Nr	Number of dialogs	Average accuracy	Standard deviation
1	33	0.8576	0.0582
2	52	0.8415	0.0469
3	68	0.8536	0.061
4	82	0.8712	0.0617
5	97	0.8919	0.0351
6	122	0.8492	0.0378
7	128	0.8468	0.0309
8	160	0.8617	0.0231

We have trained the model several times by increasing the number of dialogs in training data. The first set of dialogs contains conversations where the bot trainer played both roles (in 48.75 % of collected dialogs) until the set of actions and entities was adjusted for the domain. Further conversations are between the bot trainer and different users (51.25 % of collected dialogs).

The models are tested using 10-fold cross-validation. See **Table 2** for the average accuracy and standard deviation (among 10 folds) of each model.

The average accuracy for the neural model trained with collected data is 0.86 with the Standard deviation 0.0231. These results are close to those we obtained in experiments with DSTC data.

5. Conclusion

In this paper, we described experiments with different training data encodings to train the next dialog action prediction model with higher accuracy. To choose the architecture of the model, the experiments were conducted with the corpus of DSTC dialogs available in English. A variety of sentence embedding algorithms and a different type of entity representation were tested. For further experiments, the architecture of the recurrent neural network with LSTM cells was selected. The training data contains information about sentence embedding, the entities used in a dialog turn, and the history of the entities set in the previous turns of a dialog. As the embedding vector is created for a whole sentence, the effectiveness of the chosen method does not depend on the length of each sentence.

The selected architecture was used to train the next dialog action prediction model using Latvian dialog data. The results are similar. The accuracy for the DSTC model is 0.84, and for the Latvian data, it is 0.86. This verifies that the method is language independent. There are only 160 dialogs in the Latvian dialog set, whereas there are approximately 5,500 in the DSTC corpus. DSTC dialogs are very similar – the user has to choose the type of food, restaurant location, price category, and a few more parameters. It can be concluded that it is not necessary to have such a large number of similar dialogs for training.

6. Acknowledgments

The research leading to these results has received funding from the research project "Competence Centre of Information and Communication Technologies" of EU Structural funds, contract No. 1.2.1.1/18/A/003 signed between IT Competence Centre and Central Finance and Contracting Agency, Research No. 2.3 "Neural network machine learning techniques for automated creating of virtual assistant dialog scenarios".

References

[1] Deksne D, Vasiļjevs A. Collection of Resources and Evaluation of Customer Support Chatbot. In: Muischnek K, Müürisep K, editors. Human Language Technologies – The Baltic Perspective: Proceedings of the Eighth International Conference Baltic HLT 2018 (Vol. 307). 2018 Sept 27-29; Tartu, Estonia. IOS Press; c2018. p. 30-37.

[2] Jin X, Lei W, Ren Z, Chen H, Liang S, Zhao Y, Yin D. Explicit State Tracking with Semi-Supervision for Neural Dialogue Generation. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management; October 2018; Torino, Italy. New York (NY): ACM; c2018. p. 1403-1412.

[3] Li J, Monroe W, Ritter A, Jurafsky D, Galley M, Gao J. Deep Reinforcement Learning for Dialogue Generation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. c2016. p. 1192-1202.

[4] Liu B, Tür G, Hakkani-Tür D, Shah P, Heck L. Dialogue Learning with Human Teaching and Feedback in End-to-End Trainable Task-Oriented Dialogue Systems. In: Proceedings of the 2018 Conference of

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)(Vol. 1). c2018. p. 2060-2069.

[5] Mo K, Zhang Y, Li S, Li J, Yang Q. Personalizing a dialogue system with transfer reinforcement learning. In: McIlraith SA, Weinberger KQ, editors. Thirty-Second AAAI Conference on Artificial Intelligence. AAAI Press; 2018, April. p. 5317-5324.

[6] Wen TH, Vandyke D, Mrksic N, Gasic M, Rojas-Barahona LM, Su PH., ... & Young S. A network-based end-to-end trainable task-oriented dialogue system. arXiv preprint arXiv:1604.04562. 2016.

[7] Williams J, Zweig G. Deep learning of bots through examples and experience. U.S. Patent Application No. 15/154,017. 2017.

[8] Williams JD, Zweig G. End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning. arXiv preprint arXiv:1606.01269. 2016.

[9] Eric M, Krishnan L, Charette F, Manning CD. Key-Value Retrieval Networks for Task-Oriented Dialogue. In: Jokinen K, Stede M, DeVault D, Louis A, editors. Proceedings of the the 18th Annual SIGdial Meeting on Discourse and Dialogue; 2017 Aug 15-17; Saarbrücken, Germany. ACL; c2017. p. 37-49.

[10] Lowe R, Pow N, Serban I, Pineau J. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In: Proceedings of the SIGDIAL 2015 Conference, The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 2-4 September 2015, Prague, Czech Republic. ACL; c2015. p. 285-294.

[11] Chen H, Ren Z, Tang J, Zhao EY, Yin D. Hierarchical variational memory network for dialogue generation. In: Proceedings of the 2018 World Wide Web Conference on World Wide Web, c2018. p. 1653-1662.

[12] Budzianowski P, Wen TH, Tseng BH, Casanueva I, Ultes S, Ramadan O, Gasic M. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018. Association for Computational Linguistics; c2018. p. 5016-5026.

[13] Kingma D, Ba J. Adam: A method for stochastic optimization. In: Bengio Y, LeCun Y, editors. Proceedings of 3rd International Conference on Learning Representations; 2015 May 7-9; San Diego, CA, USA.

[14] Bojanowski P, Grave E, Joulin A, Mikolov T. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics. 2017; 5:135-146.

[15] Balodis K, Deksne D. Intent detection system based on word embeddings. In: Agre G, van Genabith J, Declerck T, editors. 18th International Conference on Artificial Intelligence: Methodology, Systems, and Applications; 2018 Sept 12-14; Varna, Bulgaria. Springer, Cham; c2018. p. 25-35.

[16] Devlin J, Chang MW, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. 2018.