Leveraging BERT for Natural Language Understanding of Domain-Specific Knowledge

Vasile Ionut Iga
Business Informatics Research Center
Babes-Bolyai University
Cluj-Napoca, Romania
vasile.iga@stud.ubbcluj.ro

Gheorghe Cosmin Silaghi
Business Informatics Research Center
Babes-Bolyai University
Cluj-Napoca, Romania
Gheorghe.Silaghi@ubbcluj.ro

***Abstract*—Natural Language Understanding is a core task when building conversational agents, fulfilling the objectives of understanding the user's goal and detecting any valuable information regarding it. NLU implies Intent Detection and Slot Filling, to semantically parse the user's utterance. One caveat when training a Deep Learning model for a domain specific NLU is the lack of specific datasets, which leads to poorly performing models. To overcome this, we experiment with fine-tuning BERT to jointly detect the user's intent and the related slots, using a custom-generated dataset built around a organization specific knowledge base. Our results show that well-constructed datasets lead to high detection performances and the resulting model has the potential to enhance a future task-oriented dialogue system.**

***Index Terms*—natural language understanding, intent detection, slot filling, BERT, task oriented dialogue system**

I. Introduction

Artificial Intelligence became a hot topic domain in recent years, and the ability to understand natural language is a true requirement for such systems [1]. Conversational agents try to implement this ability, in the form of chatbots or task-oriented dialogue systems. Nonetheless, each agent solves the task of understanding the user's utterance by enhancing the Natural Language Understanding module. NLU covers Intent Detection (ID) and Slot Filling (SF); the former deals with recognizing the user's goal from a sentence, while the latter identifies key information accompanying it. Treated as two separate goals, recent literature presents neural networks that jointly solve them, as they are and highly correlated and tied [2]. Recent advances [3]–[7] make use of transfer learning to construct ID and SF classifiers starting from pre-trained models such us BERT [8], T5 [9], or GPT-3 [10].

Task-oriented dialogue (TOD) systems help users to solve particular tasks related with a specific context [11], [12]. That means tailoring all constructed models for the domain knowledge describing that context. Our long term goal is to construct a TOD system for automating tasks within a business organization characterized by a knowledge base (KB), persisted in a machine-readable format. Therefore, we can leverage this knowledge base for producing the requested dataset for training the NLU component of a future TOD system. Initially, our TOD system [13] provides support with the essential Create-Retrieve-Update-Delete (CRUD) procedures for the management of the organizational KB. Therefore, the NLU module needs to identify the intents related with those procedures and the associated slots, strongly tied with the specific instances stored within the organizational KB.

Our work focuses on fine-tuning a BERT instance to jointly detect intent and associated slots, trained using a custom-generated dataset for recognizing domain-specific actions. We assess the quality of the simple architecture built on top of BERT by first separately fine-tuning it on two popular datasets, ATIS [14] and SNIPS [15], and compare results with other state-of-the-art BERT models for NLU. The custom-generated dataset was built using a dialogue simulator [13], that exploits a given organizational knowledge base and generates dialogues between two machines, simulating human-like discussions regarding the management of the KB. Results show that minimal adjustments have to be done to pre-trained models like BERT, if the given dataset is properly fitting the desired domain, leading to high model performance.

The paper is structured as follows: section II presents related work that influenced our experimental research, section III shows our methodology, including the dataset, model's architecture, performance evaluation procedure and metrics. Section IV discusses the obtained results and sectionV concludes the paper.

The present work received financial support through the project: *Integrated system for automating business processes using artificial intelligence*, POC/163/1/3/121075 - a project co-financed by the European Regional Development Fund (ERDF) through the Competitiveness Operational Programme 2014-2020.

V.I. Iga thanks Babeş-Bolyai University for granting him the special scholarship for scientific activity during the academic year 2022-2023.

Computational resources were provided by the high-performance computational facility of the Babeş-Bolyai University (projects 48801/1862 and POC/398/1/1/124155) co-financed by the European Regional Development Fund of the European Union.

II. Related work

ID can be thought of as a sentence classification problem, while SF is considered a sequence labeling task. Let's analyze the user's utterance "Insert a project with code as 123 and class as Python". A NLU model would consider "insert" as the intention, while "code = 123" and "class = Python" are the related slots. Usually, all the detectible intents and slots are defined before constructing the classification models, and

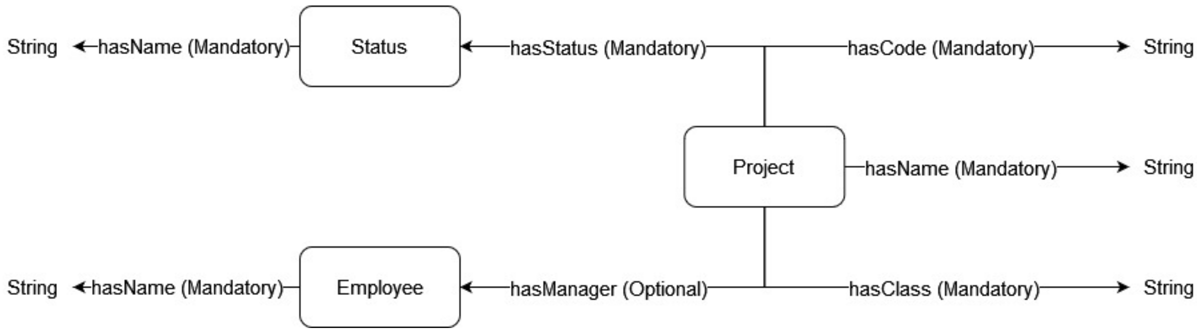

Fig. 1. The input ontology

they are fit to the target domain of the NLP problem under investigation.

Initially, ID and SF were not solved together. Ravuri and Stolcke [16] used RNN and LSTM models to detect user's intention, while Mesnil et al. [17] implemented and compared a variety of RNN architectures, including Elman and Jordan type, to detect slots. Later, the two tasks were combined to exploit their connections. Hakkani-Tur et al. [18] leveraged the power of combining bidirectional RNN with LSTM networks, to solve both tasks at the same time. Another interesting approach was introduced by Zhang et al. [19], who introduces the Graph LSTM to tackle the shortcomings of sequential models and exploit the correlation between slots and intents.

Recently, pre-trained DL models helped researchers to achieve new state-of-the-art results, by using the concept of transfer learning. This means one does not need to start from a vanilla model but can inherit the already learned knowledge and fine-tune it for its desired task. Examples of popular pre-trained models are BERT [8], T5 [9], or GPT-3 [10]. Other studied paradigms are cross-lingual [20], or low-resource NLU [21], for transferring knowledge to low-resource languages.

BERT [8] is a popular pre-trained model among researchers, therefore a large number of NLU versions were fine-tuned. Chen et al. [3] only add a softmax layer on top of BERT to predict intent and slots, but prove that it is enough to achieve great results on datasets such as ATIS or SNIPS. Castellucci et al. [4] attach two layers on top of BERT, one for intent detection and the other for slot filling, but design a single loss function for both tasks. Also, they extend their research by generating an Italian dataset and proving that training a model over multiple languages, even with noticeable syntax differences, can increase the overall performance of a model.

Zhang et al. [5] propose an encoder-decoder framework, leveraging BERT as an encoder and implementing a two-stage decoding process for ID and SF. An interesting mechanism is the insertion of the intent encoding into the encodings of each associated slot, to increase the detection of intent-specific slots and leave out the unimportant ones.

Krone et al. [6] propose a few-shot joint learning task, to improve the performance on labels not seen at training time. They prove that their algorithm, together with a pre-trained network such as BERT, is complementary and yields further gains. Qin et al. [7] modify the standard attention mechanism from vanilla transformers with a co-interactive module that considers the bidirectional connection between intents and slots, rather than the classical approach which focuses only on the intents-slots direction. Additionally, they connect their approach with BERT, proving that pre-trained models can increase the final performances of a system.

III. Methodology

In this section, we introduce the architecture of our model, the datasets, and the training and testing procedures. The code was written in Python 3.10 and computational resources were provided by high-performance computational facility of the Babeş-Bolyai University [22]. The tokenizer and the base BERT transformer are loaded using the Hugging Face libraries, while the joint NLU component is described using the Tensorflow library. We followed the suggestions of Chen et al. [3] and Shawon Ashraf[1], who implemented their models using Tensorflow.

A. Datasets

The performance of a deep learning model strongly depends on the quality of the datasets used for training. Therefore, to increase the model's capacity of recognizing entities of interest, one needs to use specific data, well covering the search space of the target problem. In general, public datasets are tailored for solving general problems, but for specific tasks, such as the one introduced by this paper, those datasets are of little help.

To accomplish our needs, i.e. creating a dataset with conversations related to our specific concepts and instances within our organization, we designed a conversation generator, described in [13]. The system generates conversations about specific concepts from a provided ontology, growing an organizational knowledge base (KB) related with the concepts supplied in the ontology. The final goal of each conversation is to perform CRUD (Create-Retrieve-Update-Delete) operations on the organizational KB, thus collecting important information regarding the target organization, meantime generating

[1]https://github.com/ShawonAshraf/nlu-jointbert-dl2021

the properly annotated dataset for the NLP tasks, in our case, intent detection and slot filling.

The starting ontology is presented in Fig. 1. It describes three concepts (Project, Status, and Employee) and the relationships between them or other literal values such as strings or integers. Each relationship represents a parameter assigned to a concept. In each conversation the user asks the system to perform one or more CRUD operation over the organizational KB and if the requested operation is confirmed by the system, it is persisted as valid in the organizational KB. Each user utterance could request one of the 14 intents, with a total of 27 slots (the specific user intents are described in detail in [13]).

Generated data is presented in the JSON format depicted in Fig.2. Each phrase has a unique ID and the key elements stored are the user utterance, intent of the phrase, important slots, and their positions.

```
{
 'utterance ID': {
        'text': 'user utterance',
        'slots': {'slot name': 'slot value' etc.}
        'positions': {'slot name': [start index, end index], etc.}
        'intent': 'detected intent'
 }...
}
```

Fig. 2. Generated dataset format

We can generate datasets with variable number of conversations, thus, we generated multiple datasets with number of conversations ranging from 625 to 5000, equivalent to 2500 to 17500 user utterances (phrases) per dataset, letting us to select the best dataset alternative considering the tradeoff between training performance and training time.

Table I presents the statistics of the generated datasets (GD) used for model training and performance assessment. In each case, the validation procedure used for model fine-tuning was the hold-out procedure with 20% of the training data kept for validation. For testing dataset we selected about 1000 conversations or a number of conversation no more that 25% of the training data.

TABLE I
STATISTICS REGARDING THE GENERATED DATASETS

Statistics / Dataset	GD1	GD2	GD3	GD4
No of conversations	625	1250	3250	5000
No of user utterances (phrases)	2458	4677	9750	17515
Training dataset (no. phrases)	2000	3800	8750	16500
Test dataset (no. phrases)	458	877	1000	1015

B. The model's architecture

The architecture of our model starts from a vanilla version of BERT, with two added layers on top of it. One dense layer is used for deciding about the intent, while another dense layer detects important slots associated with it. Fig. 3 depicts the model's architecture.

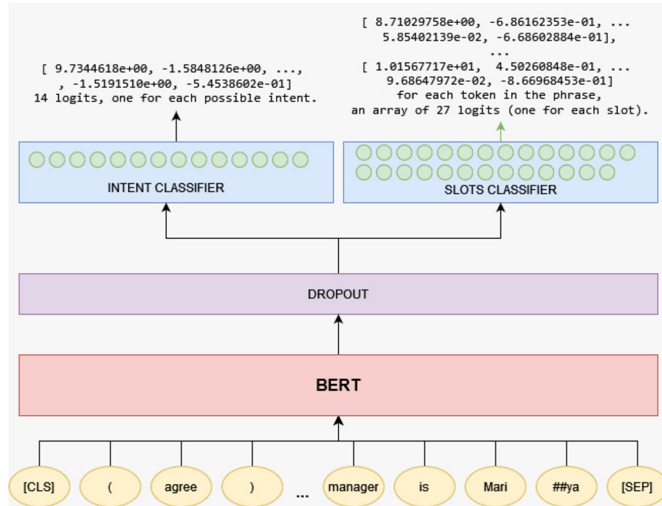

Fig. 3. Model's architecture

Text processing starts by tokenizing the input with a BERT-specific tokenizer, that uses the sub-word technique. Next, the tokens are fed into a pre-trained instance of BERT. A dropout layer with the rate of 0.1 is added in order to prevent overfitting and increase the overall performance of the system. The last step is predicting the intent and slots of the input utterance. Both classifiers contain a number of perceptrons equal to the number of intents and slots, respectively. Finally, the model outputs logit values, in the form of tensors. For intent detection, the output tensor contains 14 logit values. For slot filling, the model outputs a tensor with 27 logit values for each token in the sentence. To predict the final label, we choose the highest logit from each tensor, get its ID and convert it into natural language using a dictionary of intents/slots.

The optimizer selected for the perceptrons is Adam, with a learning rate of $3e{-}5$, and $\epsilon = 1e{-}8$, to avoid weights division by zero. Categorical cross-entropy is selected as the loss function for each task. The metric observed during training is sparse categorical accuracy. The batch size is 32 and the epochs number equals 2. All parameter values are set to respect the guidelines from [8] when dealing with fine-tuning operations of BERT for NLP tasks.

C. Model's performance evaluation

The main metrics that assess the quality of a model are intent accuracy, slot F1 score, and overall accuracy. The intent accuracy is a simple metric that tells us the percentage of times a model predicted the right intent, while the slot F1 score assures that the model keeps a high balance between precision and recall, to prevent wrong label predictions from propagating to further tasks. The overall accuracy measures whether both intent and slots were correctly predicted in one utterance, which is the desired behavior of a model.

D. Model's baseline performance

The first step of the training process was to assess the model's baseline capacity. Therefore, we have separately fine-tuned our BERT-based architecture presented in subsection III-B on ATIS [14] and SNIPS [15], comparing the results with other state-of-the-art BERT models.

ATIS consists of transcripts of audio recordings between humans and some automated airline travel inquiry systems. The training dataset contains 4478 utterances, while the test set has 893. It has 120 slot labels and 21 intent types in the training data. SNIPS includes phrases generated by the interaction between humans and a smart assistant that can execute a variety of desired tasks. The train set has 13084 phrases, while the test set consists of 700. There are 72 slot labels and 7 intent types.

The training procedure for ATIS and SNIPS is not straightforward as our data format is JSON, while ATIS and SNIPS data are texts. JSON objects need to have unique names for each property, thus a label can only appear once in a phrase. This represents a compatibility problem between the JSON format and simple texts of ATIS and SNIPS, that lead us to restrict each label to a maximum of one appearance per user utterance. While most of the labels in the ATIS and SNIPS datasets already had only one appearance per phrase, only little amount of information was lost during the conversion of ATIS and SNIPS to JSON, making the direct comparison between models still possible.

Table II presents the performance of our fine-tuned BERT-based architecture for intent detection and slot filling on ATIS and SNIPS datasets. We notice that our model yields good results on the individual tasks, but low overall accuracy. The loss of information induced by the conversion from text to JSON is the cause for this low overall accuracy. Nonetheless, our proposed architecture seems promising to solve our specific intent detection and slot filling tasks.

TABLE II
JOINT MODEL PERFORMANCE ON INTENT DETECTION AND SLOT FILLING FOR BERT-BASED MODELS

	ATIS [14]			SNIPS [15]		
Model	Intent Acc	Slot F1	Overall Acc	Intent Acc	Slot F1	Overall Acc
BERT-SLU [5]	99.76	98.75	93.89	98.96	98.78	96.76
Co-interactive transformer + BERT [7]	98.00	96.10	88.80	98.80	97.10	93.10
Joint BERT [3]	97.50	96.10	88.20	98.60	97.00	92.80
BERT-Joint [4]	97.80	95.70	88.20	99.00	96.20	91.60
NLU BERT (our work)	97.31	87.85	55.20	97.57	88.97	62.00

IV. RESULTS

In this section we present the results of the model fine-tuned on the generated datasets presented in subsection III-A, with the above-described methodology.

TABLE III
PERFORMANCE METRICS FOR MODELS FINE-TUNED ON GENERATED DATASETS WITH VARIOUS NUMBER OF CONVERSATIONS

Metrics / Training Dataset	GD1	GD2	GD3	GD4
Training intent accuracy	0.9862	0.9967	0.9994	1
Validation intent accuracy	0.9900	1	0.9983	1
Training slot accuracy	0.9495	0.9704	0.9925	0.9977
Validation slot accuracy	0.9588	0.9846	0.9943	0.9996
Time to train (min.sec)	1.49	2.28	5.55	10.04
Test intent accuracy			0.9990	
Test overall slot F1 score			0.9525	
Test overall accuracy			0.8860	

First, we fine-tuned our architecture for the generated datasets described by table I. Results are presented in table III.

Training happens surprisingly quick, as for a dataset of 7000 instances (GD3) it took 5 minutes and 55 seconds to finish the 2 training epochs. Increasing the number of instances in the dataset leads to a more than proportional increase in the training time. Only the model trained on the smallest dataset (GD1) seems to be under-trained, with a intent detection accuracy of about 98% and slot detection accuracy of about 95%. Models trained on big enough number of conversations are good candidates for selection, as the intent and slot detection accuracy overpass 99%. Because the model trained on GD3 already reached 99% accuracy and it's training time seems reasonable (about half than the next candidate), we selected it as our final model, having the best accuracy / time-to-train ratio.

On the test set, the selected model successfully predicted almost all the intents (99.9% accuracy), yielded a 95.25% overall slot F1 score, and kept a balanced overall accuracy of 88.6%. This proves that a powerful pre-trained model, such as BERT, can be fine-tuned with a well-designed dataset to obtain excellent results, leading to the possibility of integrating it into a conversational agent's architecture.

Table IV presents an extended analysis of the performance achieved by the model for each slot. Each line of the table presents the performance for a given slot of the user utterance. *new values* refer to novel values proposed by the user for a given slot, *old values* refer to slot values corresponding either to instances already existing in the organizational knowledge base or values that the user tries to search (or retrieve) from the existing KB.

Within our general TOD architecture, correctly detecting the *entity type* and the *instance* is crucial. These let the system to properly locate the corresponding relevant information in the KB that accompanies our TOD system. We indeed achieve this, as performance metrics for *entity type* are almost equal to 1, and the recall for *instance* is 1.

The model is able to successfully detect the parameter slots (*hasClass*, *hasCode*, *hasName*, *hasManager*, *has Status*, and *hasRole* - see Fig. 1), crucial for the correct insertion / retrieval of instances in the organizational knowledge base.

The *new values* and *old values* slots were more difficult to detect. These slots may appear only in update requests, where the user wants to update existing instances that fit certain

TABLE IV
TEST DATASET PERFORMANCE METRICS FOR EACH SLOT

Slot / Metric	Precision	Recall	F1 score
entity type	1	0.9980	0.9990
instance	0.9235	1	0.9602
hasClass	0.9499	0.9768	0.9631
hasName	0.9723	0.9832	0.9777
hasCode	0.9550	0.9749	0.9645
hasStatus	0.9620	0.9731	0.9676
hasManager	0.9829	0.9791	0.981
hasRole	0.9633	1	0.9813
new values hasClass	0.8046	0.8140	0.8092
new values hasName	0.8111	0.9605	0.8795
new values hasCode	0.8261	0.8352	0.8306
new values hasStatus	0.5106	0.9411	0.6621
new values hasManager	0.9568	0.9595	0.9582
new values hasRole	0.9082	0.957	0.9319
old values hasClass	0.6792	0.9730	0.8
old values hasName	0.6786	1	0.8085
old values hasCode	0.6923	0.9183	0.7895
old values hasStatus	0	0	0
old values hasManager	0.6883	0.9464	0.797
old values hasRole	0.7808	1	0.8769

filters (hence, *old values*) with new information for some slots (hence, *new values*). The property of slot value being *new* or *old* is differentiated by the presence of other words in the utterance, thus the model has a harder job in identifying those values, and this ends in a lower performance. Most of these slots have significantly higher recall than precision, underlying the fact that the model misplaced values between *new* and *old* labels. A deeper analysis confirmed our conclusion, as for the *old values* of *hasStatus* parameter the metrics are equal to 0. The model predicted the *new values* for *hasStatus* label for all *old values* of *hasStatus* instances, therefore this explains the low metrics.

V. Conclusion

In this paper, we have successfully proven that a popular pre-trained model such as BERT, with minimal architectural additions, can solve the task of NLU on a custom-generated dataset. We have tested the proposed architecture on well-known datasets for intent detection and slot-filling tasks, such as ATIS and SNIPS, to establish a baseline verification of the architectural effectiveness. The in-depth analysis of the performance metrics for each slot in the test split has shown that the model can yield high performance, when sufficient number of instances were seen during training.

Future work will focus on better-generated datasets, that may contain several variations of phrases to include more instances for each slot. Also, we will work on replacing the JSON structure to enable multiple same-slot appearances in a single phrase and align ourselves with standard ATIS or SNIPS formats. Finally, the model will be included in a task-oriented dialogue system, to solve the NLU task and we will test its performance in real-life scenarios.

References

[1] D. Adiwardana, M. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, and Q. V. Le, "Towards a human-like open-domain chatbot," *CoRR*, vol. abs/2001.09977, 2020. [Online]. Available: https://arxiv.org/abs/2001.09977

[2] L. Qin, T. Xie, W. Che, and T. Liu, "A survey on spoken language understanding: Recent advances and new frontiers," in *30th International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Z. Zhou, Ed. ijcai.org, 2021, pp. 4577–4584. [Online]. Available: https://doi.org/10.24963/ijcai.2021/622

[3] Q. Chen, Z. Zhuo, and W. Wang, "BERT for joint intent classification and slot filling," *CoRR*, vol. abs/1902.10909, 2019. [Online]. Available: http://arxiv.org/abs/1902.10909

[4] G. Castellucci, V. Bellomaria, A. Favalli, and R. Romagnoli, "Multi-lingual intent detection and slot filling in a joint BERT-based model," *CoRR*, vol. abs/1907.02884, 2019. [Online]. Available: http://arxiv.org/abs/1907.02884

[5] Z. Zhang, Z. Zhang, H. Chen, and Z. Zhang, "A joint learning framework with BERT for spoken language understanding," *IEEE Access*, vol. 7, pp. 168 849–168 858, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2954766

[6] J. Krone, Y. Zhang, and M. Diab, "Learning to classify intents and slot labels given a handful of examples," in *2nd Workshop on Natural Language Processing for Conversational AI*. Association for Computational Linguistics, 2020, pp. 96–108. [Online]. Available: https://aclanthology.org/2020.nlp4convai-1.12

[7] L. Qin, T. Liu, W. Che, B. Kang, S. Zhao, and T. Liu, "A co-interactive transformer for joint slot filling and intent detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 2021, pp. 8193–8197. [Online]. Available: https://doi.org/10.1109/ICASSP39728.2021.9414110

[8] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://doi.org/10.18653/v1/n19-1423

[9] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[10] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[11] H. Chen, X. Liu, D. Yin, and J. Tang, "A survey on dialogue systems: Recent advances and new frontiers," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 2, pp. 25–35, 2017. [Online]. Available: https://doi.org/10.1145/3166054.3166058

[12] Z. Li, H. Wang, A. Albalak, Y. Yang, J. Qian, S. Li, and X. Yan, "Making something out of nothing: Building robust task-oriented dialogue systems from scratch," in *Alexa Prize TaskBot Challenge Proceedings*, 2022. [Online]. Available: https://www.amazon.science/alexa-prize/proceedings/making-something-out-of-nothing-building-robust-task-oriented-dialogue-systems-fro

[13] V. I. Iga and G. C. Silaghi, "Ontology-based dialogue system for domain-specific knowledge acquisition," in *31st International Conference on Information Systems Development, ISD 2023, Lisbon, Portugal, August 30 - September 1, 2023*. AIS Library, 2023, to appear.

[14] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, USA, June 24-27, 1990*. Morgan Kaufmann, 1990. [Online]. Available: https://aclanthology.org/H90-1021/

[15] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril, M. Primet, and J. Dureau, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," *CoRR*, vol. abs/1805.10190, 2018. [Online]. Available: http://arxiv.org/abs/1805.10190

[16] S. V. Ravuri and A. Stolcke, "Recurrent neural network and LSTM models for lexical utterance classification," in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*. ISCA, 2015, pp. 135–139. [Online]. Available: https://doi.org/10.21437/Interspeech.2015-42

[17] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013*, F. Bimbot, C. Cerisara, C. Fougeron, G. Gravier, L. Lamel, F. Pellegrino, and P. Perrier, Eds. ISCA, 2013, pp. 3771–3775. [Online]. Available: https://doi.org/10.21437/Interspeech.2013-596

[18] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y. Chen, J. Gao, L. Deng, and Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM," in *INTERSPEECH 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, N. Morgan, Ed. ISCA, 2016, pp. 715–719. [Online]. Available: https://doi.org/10.21437/Interspeech.2016-402

[19] L. Zhang, D. Ma, X. Zhang, X. Yan, and H. Wang, "Graph LSTM with context-gated mechanism for spoken language understanding," in *34th AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 9539–9546. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6499

[20] W. Xu, B. Haider, and S. Mansour, "End-to-end slot alignment and recognition for cross-lingual NLU," in *2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020, pp. 5052–5063. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-main.410

[21] Y. Hou, W. Che, Y. Lai, Z. Zhou, Y. Liu, H. Liu, and T. Liu, "Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network," in *58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020, pp. 1381–1393. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.128

[22] D. Bufnea, V. Niculescu, G. Silaghi, and A. Sterca, "Babes-Bolyai University's high performance computing center," *Studia Universitatis Babes-Bolyai, Informatica Series*, vol. 61, no. 2, pp. 54–69, 2016.