DeepPavlov Dream: Platform for Building Generative AI Assistants

Dilyara Zharikova[1] dilyara.rimovna@gmail.com
Daniel Kornev[1] daniel@kornevs.org
Fedor Ignatov[1] ignatov.fedor@gmail.com
Maxim Talimanchuk[1]
Dmitry Evseev[1]
Kseniia Petukhova[1]
Veronika Smilga[1]
Dmitry Karpov[1]
Yana Shishkina[1]
Dmitry Kosenko[1]
Mikhail Burtsev[2] mbur@lims.ac.uk
[1]Moscow Institute of Physics and Technology, Russia
[2]London Institute for Mathematical Sciences, UK

Abstract

An open-source DeepPavlov Dream Platform is specifically tailored for development of complex dialog systems like Generative AI Assistants. The stack prioritizes efficiency, modularity, scalability, and extensibility with the goal to make it easier to develop complex dialog systems from scratch. It supports modular approach to implementation of conversational agents enabling their development through the choice of NLP components and conversational skills from a rich library organized into the distributions of ready-for-use multi-skill AI assistant systems. In DeepPavlov Dream, multi-skill Generative AI Assistant consists of NLP components that extract features from user utterances, conversational skills that generate or retrieve a response, skill and response selectors that facilitate choice of relevant skills and the best response, as well as a conversational orchestrator that enables creation of multi-skill Generative AI Assistants scalable up to industrial grade AI assistants. The platform allows to integrate large language models into dialog pipeline, customize with prompt engineering, handle multiple prompts during the same dialog session and create simple multimodal assistants.

1 Introduction

Complex AI assistants are becoming more and more widespread. As a result, interest in technology for building complex conversational interfaces has grown significantly over the last years. At the same time, most of the available systems enabling complex dialog systems development are proprietary or limited. This opens up new opportunities for open-source systems that facilitate the development of complex AI assistants.

The DeepPavlov Dream Platform[1] provides a stack of Apache 2.0-licensed open-source technologies that enable development of complex dialog systems such as enterprise AI assistants. The platform features a conversational AI orchestrator called DeepPavlov Agent to coordinate an asynchronous scalable dialog pipeline; a framework called DeepPavlov Dialog Flow Framework (DFF) to facilitate development of the multi-step skills; support for Wikidata and custom knowledge graphs; a library of modern NLP components and conversational AI skills (script-based, chit-chat, question answering (QA), and generative skills) organized into a set of distributions of multi-skill conversational AI systems; and a visual designer. These components make it possible for developers and researchers to implement complex dialog systems ranging from multi-domain task-oriented or chit-chat chatbots to voice and multimodal AI assistants suitable for academic and enterprise use cases.

The DeepPavlov Dream Platform supports integrating large language models (LLMs) into production-ready dialog systems with the help of general and custom knowledge graphs (KGs) for fact checking, pre- and post-filters for filtering out unsuitable responses, and prompt-based generation for indirect control of LLMs.

Multimodality in DeepPavlov Dream provides an opportunity to operate with images and to perform actions via APIs based on the user's commands extracted during the conversation.

In this paper, we present the DeepPavlov Dream Platform for dialog systems development. In Section 2, we compare the platform with existing competitors. Sections 3, 4 and 5 introduce the pipeline's

[1]https://github.com/deeppavlov/dream

orchestrator, distributions and other development tools. The platform's components and proposed approaches for dialog system development are described in Section 6. Section 7 presents our methodology for building prompt-based Generative AI Assistants. Approaches to managing multimodality are described in Section 8. Finally, Section 9 provides an overview of a custom dialog system development process.

2 Comparison with Competitors

In Table 1 we present a comparison of the popular Conversational AI Platforms. Unlike other solutions, we provide an open-source Apache 2.0-based, multi-skill platform that enables development of complex open-domain dialog systems. DeepPavlov Dream allows for combining different response generation methods, adding pre- and post-filters, utilizing Wikidata and custom knowledge graphs, designing custom dialog management algorithms, integrating large language models (LLMs) to production-ready dialog systems. DeepPavlov Dream also provides simple integration with load-balancing tools that is crucial for LLMs-based dialog systems in production. We are also working towards text-based and multimodal experiences like robotics.

3 Pipeline

The DeepPavlov Dream is built upon the DeepPavlov Agent, an open-source framework for orchestrating complex systems. The full dialog system pipeline of the DeepPavlov Dream is presented in Figure 1. There are four component groups — Annotators, Skills, Candidate Annotators, and Response Annotators — and two dialog management components — Skill Selector and Response Selector. Dialog State is a shared memory that contains all the information about the dialog as DeepPavlov Agent expects that services are stateless and can be run as multiple instances. There are two synchronization points — Skill Selector and Response Selector. The other services can be run in parallel, although the Agent allows dependencies between services within the group to effectively chain them.

DeepPavlov Agent's pipeline is asynchronous, i.e., one user's request does not block the agent and does not prevent the agent from receiving and processing requests from other users. Each service is deployed in a separate `docker` container. In addition, separate containers are used for the agent itself and the database. For development, one can run the dialog system locally using `docker-compose` or Kubernetes.

4 Distributions

Original DREAM Socialbot (Kuratov et al., 2020; Baymurzina et al., 2021b) included a large number of components for Natural Language Understanding (NLU) and Natural Language Generation (NLG). Different components are run independently and accept dialog state in a required format. Although some components may depend on the other ones' annotations, in general modular system's elements can be safely removed or replaced with their analogues. Platform users can re-use existing components to design their own dialog systems or develop custom components and include them into existing or custom dialog systems.

The DeepPavlov Dream Platform utilizes a distribution-based approach for dialog systems development. A *distribution* is a set of YML-files specifying parameters of `docker` containers, and a configuration JSON-file determining a processing pipeline for DeepPavlov Agent. Platform contains different distributions including script-based English distributions, generative-based English, Russian and multi-lingual distributions, multimodal distribution, robot controller distribution, and lots of multi-skill distributions utilizing prompt-based generation with LLMs (details in Section 6).

5 Development Tools

Platform is supported by development tools — `DeepPavlov dreamtools` and DF Designer (Kuznetsov et al., 2021), a visual aid in developing scenario-driven skills for the Dream Platform using DFF[2].

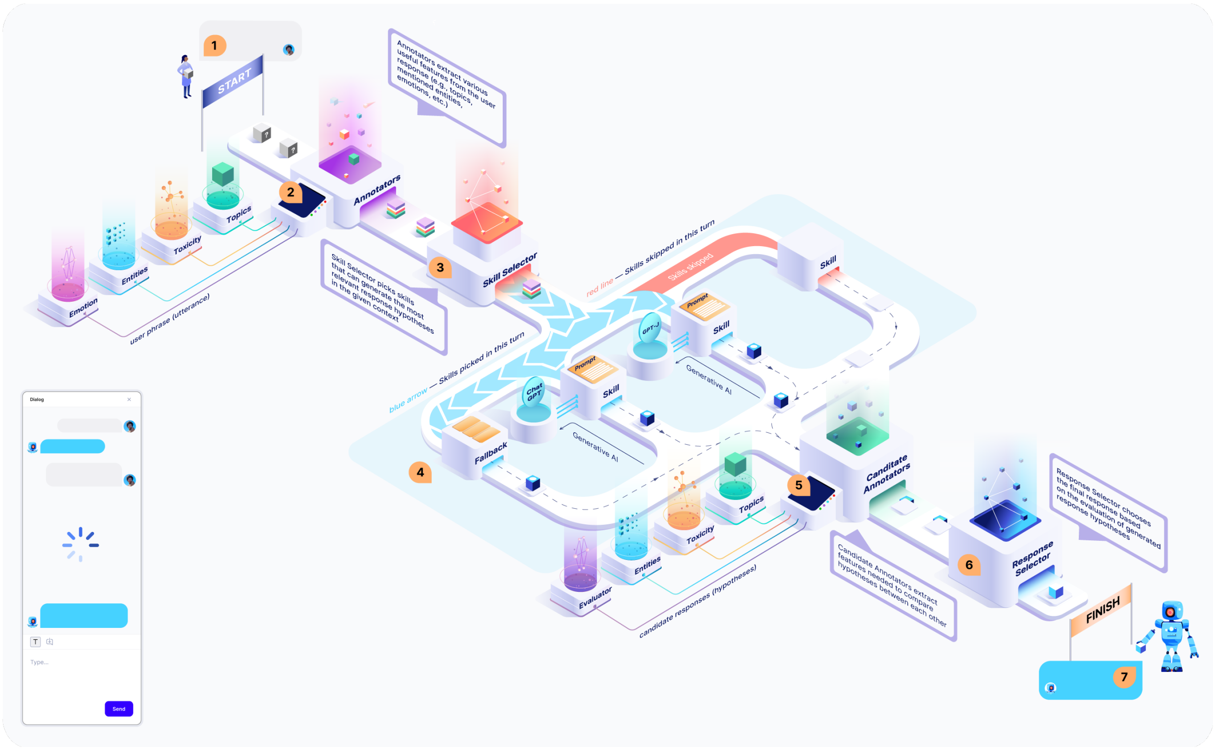
`DeepPavlov dreamtools` is a set of tools in Python with a built-in command line tool which allows developers to operate with Platform distributions in a programmatic way.

Python Package The package exposes Dream distribution API via configuration and component objects. It supports a strict set of configuration files: one pipeline JSON-file and 4 docker-compose YML configuration files (for production, development, proxy, and local deployments). For each file, the package implements configuration definitions

[2]`https://github.com/deeppavlov/dialog_flow_framework`

	DeepPavlov Dream	Mycroft AI	Linto AI	RASA	Amazon Lex	Google DialogFlow	IBM Watson	Avaamo	Kore.AI	Amelia
License	Apache 2.0	Apache 2.0	AGPL-3.0	Apache 2.0	N/A	N/A	N/A	N/A	N/A	N/A
Open Source	Yes	Yes	Yes	Yes	No	No	No	No	No	No
On-Premises	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	Yes
Multi-skill	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes
Generative AI	Yes	Limited	Limited	Limited	No	Limited	Yes	Yes	Yes	Limited
FAQ Skills	Yes	No	No	Yes	No	Yes	Yes	Yes	Yes	Yes
Task-oriented Skills	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Chit-Chat Skills	Yes	Limited	Limited	Limited	Limited	Limited	Limited	Limited	Limited	Yes
QA Skills	Wikidata, Custom	No	No	Custom	No	Custom	Custom	Custom	Custom	Custom
Knowledge Graphs	Wikidata, Custom KGs	No	No	Yes	No	Custom	Custom	Yes	Custom	Yes
Multilingual NLU	Per-lang, multilingual	Per-lang	Per-lang	Per-lang	Per-lang	Per-lang	Per-lang	Per-lang	Per-lang	Per-lang
Domain-specific NLU	No	No	Yes	3rd-party	Limited	3rd-party	Yes	Yes	Yes	Yes
External NLU	Yes	Yes	No	Yes	No	Partial	No	Yes	No	Unknown
Scalability	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Multimodality	Text,Voice-3, Image(wip), Video(wip)	Text, Voice-3	Text, Voice	Text, Voice-3	Text, Voice	Text, Voice	Text, Voice	Text, Voice-3	Text, Voice-3	Text, Voice

Table 1: Comparison of popular Conversational AI Platforms: DeepPavlov Dream (DeepPavlov, 2023), Mycroft AI (AI, 2022b), Linto AI (AI, 2022a), RASA (RASA, 2022), Amazon Lex (Amazon, 2022), DialogFlow (Google, 2022), IBM Watson (Watson, 2022), Avaamo (Avaamo, 2022), Kore.AI (Kore.ai, 2022), Amelia (Amelia, 2022). In Multimodality row Voice-3 denotes usage of third-party applications for Voice processing



Annotators extract various useful features from the user response (e.g., topics, mentioned entities, emotions, etc.)

Skill Selector picks skills that can generate the most relevant response hypotheses in the given context

Candidate Annotators extract features needed to compare hypotheses between each other

Response Selector chooses the final response based on the evaluation of generated response hypotheses

Figure 1: The high-level architecture of the user utterance's processing in DeepPavlov Dream pipeline. The DeepPavlov Dream supports any number of components limited only by the available computational resources

as objects with methods for serializing, listing, creating, editing, and validating components. The components are defined as generic structures.

Additionally, its higher-level functionality allows for creating new distributions from scratch or using the existing ones as a template, editing distributions, verifying consistency between component definitions (e.g., correct port forwarding, component naming, etc.), and creating DFF-based skills with all the necessary template files. We have also trnasferred distribution testing utilities previously bundled with Dream: an automated dialog tester and a file validation script.

CLI `DeepPavlov dreamtools` can be used as a command line utility that wraps most of high-level package functionality. One of the common uses is creating a `local.yml` configuration file, which contains instructions on whether to deploy a component locally or redirect to a proxied DeepPavlov deployment.

6 Components

The platform contains English, Russian, and multilingual annotators and skills. Skill and Response Selectors are language-agnostic in general. The Skill Selector's output may include skills not present in the running pipeline, making the Skill Selector's code reusable. There are several available algorithms for Response Selector – tag-based for script-based distributions, LLM-based and ranking-based for generative distributions.

Modular container-based architecture of the Dream platform allows for integrating components from different frameworks and with different requirements. For many conversational NLP models, DeepPavlov (Burtsev et al., 2018) library is extensively used in several annotators and skills. DeepPavlov library, an open-source conversational NLP framework that is based on PyTorch and supports huggingface `transformers`, which allows it to use various transformer-based models from huggingface Hub (Burtsev et al., 2018). Traditionally, DeepPavlov contains several variants of models: the best-performing one with the highest score, the resource-efficient one with the fastest inference time (Kolesnikova et al., 2022), and the multilingual one with the support of several languages. DeepPavlov library provides ready-for-use tools for training and inference of NLP pipelines along with appropriate `docker` images that simplify integrating and adapting new components.

Annotators solve a variety of NLP tasks: text re-writing (sentence segmentation, punctuation recovery, spelling preprocessing, coreference resolution (Le et al., 2019)), text classification (sentiment, toxicity, emotions, factoidness, topics (Sagyndyk et al., 2022; Karpov and Burtsev, 2023), dialog acts (Yu and Yu, 2019), intents and speech functions (Ostyakova et al., 2022)), token classification (NER, entity detection), knowledge retrieval (requests to external APIs, entity linking to KBs (Evseev), knowledge extraction from structured and textual KBs), text ranking (selecting most relevant prompts for conditional generation, response candidate ranking (Gao et al., 2020)).

Conversational input usually does not imply correct case sensitivity, so DeepPavlov Dream utilizes NER model (Chizhikova et al., 2023) adapted to both cased and uncased inputs. Additionally, the model is based on the Multilingual BERT (MBERT), which allows to support entity extraction in multiple languages due to the MBERT cross-lingual transferability (Konovalov et al., 2020).

Resource consumption is also one of the most challenging parts of the dialog systems development. Often, the production-level annotators quality can be reached only by NN-based models each of which may require gigabytes of GPU memory. To tackle this problem, DeepPavlov equips Dream with a multi-task learning (MTL) classifier (Karpov and Konovalov, 2023) trained to solve nine problems within a single model and decreasing GPU memory usage ninefold.

Skills in the Dream Platform define response generators. There are different types of algorithms for response generation, e.g., template-based, retrieval, and generative models. The skills that plan the dialog more than one step ahead are called scripted skills. These skills are able to get the dialog to develop in depth, which is already a generally accepted expectation of users from conversational systems. The scripts may utilize either slot-filling (Baymurzina et al., 2021a) in template-based responses or controllable generation via LLMs. Creating script-based skills by hands is a labour-consuming task, so we also researched approaches for automatic scripts generation (Kapelyushnik et al., 2022; Evseev et al.). A dialog system's behavior may need to be deterministic in some cases. For that, developers may utilize either an intent-based templated response skill, a FAQ skill, or prompt-based generation via LLMs.

Prompt-based response generation via LLMs, a recent trend in NLP field (Bai et al., 2022; Taylor et al., 2022; Scao et al., 2022; Biderman et al., 2023; Köpf et al., 2023; Dey et al., 2023), was reflected in the development of prompt-based skills that utilize given prompts and LLMs to respond to the current context. Developers may create a prompt-based generative distribution featuring their own prompts by copy-pasting several lines of code and configuration descriptions and selecting the generative services of interest as a parameter (more detailed instructions are provided in our tutorials and documentation). One of the main features of DeepPavlov Dream is a multi-skill support which allows a dialog system to contain and switch between several different prompts during a dialog session. More details in Section 7.

Knowledge Bases help the dialog system to generate meaningful responses that require world knowledge. Although generative models may learn some information from the training data, correct responses to factoid question require the dialog system to utilize an updating Knowledge Base. Knowledge Bases contain facts in graph or textual form which can be used as knowledge of the dialog assistant. DeepPavlov Dream utilizes graph and textual KBs for different annotators and skills, e.g., knowledge-based question answering (KBQA) (Evseev and Arkhipov, 2020), fact retrieval, paragraph-based open-domain question answering (ODQA). For AI assistants, one of the main required features is a custom ontology and a knowledge graph which allows to extract and store structured information about users. DeepPavlov Dream now integrates support of the custom KGs that allows to integrate corporate knowledge locally without any concerns on the data safety.

7 Prompt-based Generative AI Assistants

Since the emergence of LLMs, they have been used to tackle a variety of natural language tasks. The earlier and smaller models had to be fine-tuned for each specific task. However, very large models have shown a remarkable capacity of handling the same tasks few-shot and zero-shot using prompts, which are tokens appended or prepended to the model's as an instruction to "guide" its behavior.

When using LLMs in dialog modeling, it is crucially important to avoid insensitive or potentially harmful content and tailor the responses for the specific user needs. That is where prompt engineering allows to steer the model in the right direction with no need for fine-tuning.

DeepPavlov Dream provides an approach to building Generative AI Assistants with prompt-based control and lets the user develop their own ones with the use of prompt engineering. For that, the platform features Generative Skills, which encapsulate Generative Model Services used to locally run the LLMs of the user's choice or utilize external Generative API services. Each Generative Skill is controlled by one user-specified prompt, utilizes selected Generative Service and is built using Dialog Flow Framework to provide an opportunity for the developer to control the skill in a script-based manner. To build an assistant, which comes in a form of a custom Generative AI Assistant Distribution, several Generative Skills can be combined with help of Prompt Selector picking the most relevant prompts among presented and Response Selector managing the dialog. Prompt selection could be performed in different ways: simple ranking of prompt-context pairs, ranking of pairs of context and prompts goals extracted from prompts using LLMs, predicting with LLMs based on prompts descriptions.

In upcoming releases, we plan to enhance our skills by incorporating structured, vector-based, and RMT-based (Bulatov et al., 2022, 2023) representations of episodic and working memory based on the dialog state and conversation history stored within DeepPavlov Dream. Given the importance of supporting queries over documents and knowledge bases, future versions of DeepPavlov Dream will support LLM-driven interaction with external data sources like corporate databases. We also plan to add support for prompt-chaining to enable reasoning simulation to address more complex problems through planning and to facilitate development of the autonomous AI agents.

8 Multimodal Generative AI Assistants

Users' expectations from chatbots are rapidly increasing, so *multimodality*, which is operating with images, audio and video, is becoming an important direction in dialog systems' development. While the audio input can be converted to text almost without losing sense (except of intonations and emotions), received images may bring a key information to a dialog.

DeepPavlov Dream utilizes stateless paradigm, which means that components do not store any information about the dialog. The dialog state contains all the information and is forwarded through the full pipeline (partially, according to the given formatters) by the Agent component. Sending images or video between containers can be time-consuming, so we offer to use a special database storing images and videos and send file paths in this database through the pipeline.

We implemented one of the most obvious ideas, which is to convert images to textual modality using image captioning models. Textual input replacement with an original image caption does not work as expected, so we created a scripted approach which extracts topics and entities from the caption, detects the type of the objects, and returns scripted responses for several particular types of objects, such as people, animals and food. We are also working on the skills responding to the user with images to provide a more engaging and human-like experience. In addition to that, we are working on enabling Socratic Models-like (Zeng et al., 2022) approach to address user tasks by running conversations between skills understanding different modalities.

9 Developer Experience

DeepPavlov Dream Platform enables developers to build their own multi-skill dialog systems. Despite of low-code/no-code trends for development tools, our platform requires its users to work with `python` and `docker` that will give them enough flexibility and customization opportunities.

Developer's path starts with repository on GitHub[3]. Developer needs a PC with Ubuntu or other Unix-like OS capable of running `docker`. To run heavy NLP components, locally dedicated GPUs are necessary. Any IDE can be utilized, however, VS Code is needed to use our DF Designer to create custom scenario-driven skills in a visual interface.

To create a custom distribution, developer can either make it by hands or use a single command from `DeepPavlov dreamtools` specifying a list of selected components. Developer can talk to the system by running a chat in a developer's mode in a command line, via Telegram or using web interface on "/chat" endpoint of the agent[4].

One can utilize some components via proxy. Agent and database components are always run locally, while proxied components are light-weight containers. All components not present in the proxy YML-file for the current distribution are run locally, which allows the developer to run and debug any new or existing component locally.

Here are the main opportunities for customizing a dialog system in DeepPavlov Dream: one may combine a new dialog system with particular existing components, change the parameters of these components in configuration files, create new components from scratch or change the existing components.

To create a custom scenario-driven DFF skill and add it to the distribution, the developer can use a single command from `DeepPavlov dreamtools`. To visually design a custom scenario-driven DFF skill, the developer can use DF designer (refer to the Workshop video[5] for detailed instructions).

Prompt-based Generative AI Assistants can be created by hands using a template distribution. One can add any number of prompt-based skills and customize them by prompt engineering and selecting Generative Services of interest. Multi-skill dialog management is handled automatically.

Debugging a complex distributed platform is always a challenge. In the DeepPavlov Dream Platform, one can use different techniques to successfully debug their Distribution. One can utilize DeepPavlov Agent's console to debug based on the entire dialog state, docker logging output to debug individual components, or POST requests to DeepPavlov Agent via Postman or similar tools.

The developer can build a custom user experience around their Dream-based multi-skill AI assistant or connect it to some of the existing channels, such as Amazon Alexa, to make it available for the end users. There is a Workshop Video[6] available with detailed instructions.

We also provide the documentation site for DeepPavlov Dream[7]. The site provides access to comprehensive resources for building intelligent conversational assistants tailored to users' specific needs. The site offers extensive documentation, release notes, and detailed examples to facilitate the development of advanced conversational AI applications. The site also provides opportunities to join the community of developers leveraging DeepPavlov Dream to shape the future of conversational AI technology.

10 Conclusion

As complex conversational systems are becoming more and more popular, it is important to make the development process of such systems easier for researchers and developers. DeepPavlov Dream is

[3] `https://github.com/deeppavlov/dream`
[4] `http://0.0.0.0:4242/chat`
[5] `https://www.youtube.com/watch?v=WVlFV9VBh1g`
[6] `https://www.youtube.com/watch?v=WAN_I1O-M4M`
[7] `https://dream.deeppavlov.ai/`

an open-source conversational platform designed to let the users develop their own complex dialog systems and access existing NLP components for feature extraction and classification of user utterances. The platform features a variety of skills developed with scenario-driven, retrieval, and generative approaches using modern NLP techniques including prompt-based generation. These skills are organized into ready-to-use distributions of the multi-skill AI assistants. DeepPavlov Dream allows to customize dialog systems at all levels. The platform was battle-tested during Amazon Alexa Prize 3 and 4 and is now providing core infrastructure to facilitate the development of multi-skill generative AI assistants for industry and academia.

We built a DeepPavlov Dream[8] Platform's website[9] including documentation, tutorials and useful links, chat with the demo distribution. We have published a series of articles about DeepPavlov Dream on Medium[10]. We also have a YouTube channel[11] where we publish video workshops and seminars. To communicate with our team, one can use our forum[12]. Short demo video is available on YouTube[13].

Acknowledgements

We express gratitude to all of current and past developers and researchers of DeepPavlov.ai for their contribution to the DeepPavlov Dream Platform. We are thankful to Denis Kuznetsov for providing DeepPavlov Dream with the state-machine-based skill engine powered by Dialog Flow Framework. We are also grateful to Anastasia Klowait and Mikhail Zamyatin for the DeepPavlov Dream web-site, and designer Irina Nikitenko for Figure 1. Special thanks to Yurii Kuratov and Idris Yusupov who made a major contribution to the development of the original DREAM Socialbot during the Alexa Prize Challenge 3 (Kuratov et al., 2020).

Limitations

The DeepPavlov Dream Platform contains English, Russian, and multilingual components. However, the multilingual generative model does not always respond in the same language.

DeepPavlov Dream covers open-domain hybrid conversational systems able to chat on any topic in addition to the commercial scenario-driven template-based chatbots. Therefore, we propose to use generative models. Generative-based distributions are able to respond to most of the dialog contexts with various replies while consume significant computational resources requiring at least a single GPU to run models like DialoGPT. Current distributions have a limited number of task-oriented skills, like Factoid QA and weather skill.

Ethics Statement

(1) This material is the authors' own original work, which at this stage of project development has not been previously published elsewhere. (2) The paper is not currently being considered for publication elsewhere. (3) The paper reflects the authors' own research and analysis in a truthful and complete manner. (4) We acknowledge that the use of the generative language models like DialoGPT and others in production might lead to potential harm to the end user experience; while we have adopted measures to prevent inappropriate language output we can not guarantee that the dialog systems that incorporated generative models can be free of inappropriate language. (5) All conversations users have with the publicly deployed English distribution of DeepPavlov Dream available at[14] are recorded and are available for the conversational AI researchers via[15] as an open-source dialog dataset. Users have to agree to a privacy agreement prior to talking to the Dream distribution. They are warned that their dialogs will become publicly available as part of the dataset and are strongly encouraged to never share personal details with the Dream distribution. (6) The dialogs that developers have with the Dream distribution running completely via proxy (including MongoDB instance used by DeepPavlov Agent) will also be available publicly, while the dialogs between the developer and distributions that run at least DeepPavlov Agent with MongoDB instance locally will not become available as a part of our open-source conversational AI dataset.

References

Linto AI. 2022a. Linto ai. https://linto.ai/.

[8] https://github.com/deeppavlov/dream
[9] https://dream.deeppavlov.ai/
[10] https://medium.com/deeppavlov
[11] https://www.youtube.com/c/DeepPavlov
[12] https://forum.deeppavlov.ai/
[13] https://youtu.be/1EwiqNsfomI
[14] https://dream.deeppavlov.ai/
[15] http://deeppavlov.ai/dream/datasets/

Mycroft AI. 2022b. Mycroft ai. https://mycroft.ai/.

Amazon. 2022. Amazon lex. https://docs.aws.amazon.com/lex/index.html.

Amelia. 2022. Amelia. https://amelia.ai/conversational-ai/.

Avaamo. 2022. Avaamo. https://avaamo.ai/conversational-ai-platform/.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

D Baymurzina, Yu Kuratov, D Kuznetsov, D Kornev, and M Burtsev. 2021a. Evaluation of conversational skills for commonsense.

Dilyara Baymurzina, Denis Kuznetsov, Dmitry Evseev, Dmitry Karpov, Alsu Sagirova, Anton Peganov, Fedor Ignatov, Elena Ermakova, Daniil Cherniavskii, Sergey Kumeyko, et al. 2021b. Dream technical report for the alexa prize 4. *4th Proceedings of Alexa Prize*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*.

Aydar Bulatov, Yuri Kuratov, and Mikhail S Burtsev. 2023. Scaling transformer to 1m tokens and beyond with rmt. *arXiv preprint arXiv:2304.11062*.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091.

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yurii Kuratov, Denis Kuznetsov, et al. 2018. Deeppavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, System Demonstrations*, pages 122–127.

Anastasia Chizhikova, Vasily Konovalov, and Mikhail Burtsev. 2023. Multilingual case-insensitive named entity recognition. In *Advances in Neural Computation, Machine Learning, and Cognitive Research VI*, pages 448–454, Cham. Springer International Publishing.

DeepPavlov. 2023. Deeppavlov dream. https://dream.deeppavlov.ai/.

Nolan Dey, Gurpreet Gosal, Hemant Khachane, William Marshall, Ribhu Pathria, Marvin Tom, Joel Hestness, et al. 2023. Cerebras-gpt: Open compute-optimal language models trained on the cerebras wafer-scale cluster. *arXiv preprint arXiv:2304.03208*.

DA Evseev. Lightweight and accurate system for entity extraction and linking.

DA Evseev and M Yu Arkhipov. 2020. Sparql query generation for complex question answering with bert and bilstm-based model. In *Computational Linguistics and Intellectual Technologies*, pages 270–282.

DA Evseev, MS Nagovitsin, and DP Kuznetsov. Controllable multi-attribute dialog generation with pals and grounding knowledge.

Xiang Gao, Yizhe Zhang, Michel Galley, Chris Brockett, and Bill Dolan. 2020. Dialogue response ranking training with large-scale human feedback data. *arXiv preprint arXiv:2009.06978*.

Google. 2022. Google dialogflow. https://cloud.google.com/dialogflow.

Denis Kapelyushnik, Dilyara Baymurzina, Denis Kuznetsov, and Mikhail Burtsev. 2022. Automatic generation of conversational skills from dialog datasets. In *Advances in Neural Computation, Machine Learning, and Cognitive Research VI: Selected Papers from the XXIV International Conference on Neuroinformatics, October 17-21, 2022, Moscow, Russia*, pages 31–41. Springer.

Dmitry Karpov and Mikhail Burtsev. 2023. Monolingual and cross-lingual knowledge transfer for topic classification. *Proceedings of AINL 2023*.

Dmitry Karpov and Vasily Konovalov. 2023. Knowledge transfer in the multi-task encoder-agnostic transformer-based models ( ). *Komp'juternaja Lingvistika i Intellektual'nye Tehnologii*.

Alina Kolesnikova, Yuri Kuratov, Vasily Konovalov, and Mikhail Burtsev. 2022. Knowledge distillation of russian language models with reduction of vocabulary. *arXiv preprint arXiv:2205.02340*.

Vasily Konovalov, Pavel Gulyaev, Alexey Sorokin, Yury Kuratov, and Mikhail Burtsev. 2020. Exploring the bert cross-lingual transfer for reading comprehension. In *Dialogue-21*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. 2023. Openassistant conversations–democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*.

Kore.ai. 2022. Kore.ai. https://kore.ai/.

Yuri Kuratov, Idris Yusupov, Dilyara Baymurzina, Denis Kuznetsov, Daniil Cherniavskii, Alexander Dmitrievskiy, Elena Ermakova, Fedor Ignatov, Dmitry Karpov, Daniel Kornev, et al. 2020. Dream technical report for the alexa prize 2019. *Alexa Prize Proceedings*.

Denis Kuznetsov, Dmitry Evseev, Lidia Ostyakova, Oleg Serikov, Daniel Kornev, and Mikhail Burtsev. 2021. Discourse-driven integrated dialogue development environment for open-domain dialogue systems. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 29–51.

TA Le, MA Petrov, YM Kuratov, and MS Burtsev. 2019. Sentence level representation and language models in the task of coreference resolution for russian. *Computational Linguistics and Intellectual Technologies*, pages 364–373.

Lidiia Ostyakova, M Molchanova, Ksenia Petukhova, Nika Smilga, D Kornev, and M Burtsev. 2022. Corpus with speech function annotation: Challenges, advantages, and limitations. *Computational Linguistics and Intellectual Technologies*, pages 1129–1139.

RASA. 2022. Rasa platform. https://rasa.com/product/rasa-platform/.

Beksultan Sagyndyk, Dilyara Baymurzina, and Mikhail Burtsev. 2022. Deeppavlov topics: Topic classification dataset for conversational domain in english. In *Studies in Computational Intelligence, Springer*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.

IBM Watson. 2022. Ibm watson assistant. https://www.ibm.com/products/watson-assistant.

Dian Yu and Zhou Yu. 2019. Midas: A dialog act annotation scheme for open domain human machine spoken conversations. *arXiv preprint arXiv:1908.10023*.

Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. 2022. Socratic models: Composing zero-shot multimodal reasoning with language.