

REBUILD

ICT-enabled integration facilitator and life rebuilding guidance

Project start date: 01/01/2019 | Duration: 36 months

Deliverable: D4.4 Designing the digital companion

DUE DATE OF THE DELIVERABLE: 30-06-2020
ACTUAL SUBMISSION DATE: 31-07-2020

Project	REBUILD – ICT-enabled integration facilitator and life rebuilding guidance
Call ID	H2020-SC6-MIGRATION-2018-2019-2020 – DT-MIGRATION-06-2018
Work Package	<i>WP4 – DESIGNING THE DIGITAL COMPANION</i>
Work Package Leader	<i>Full Partner Name of the WP leader</i>
Deliverable Leader	<i>Full Partner Name of the partner responsible for the Deliverable</i>
Deliverable coordinator	Theodoros Semertzidis (CERTH) - (theosem@iti.gr)
Deliverable Nature	Report
Dissemination level	Public (PU)
Version	1.0
Revision	Final

DOCUMENT INFO

AUTHORS

Author name	Organization	E-Mail
Theodoros Semertzidis	CERTH	theosem@iti.gr
Nikolaos Konstantinou Panagiotis Stalidis Michalis Lazaridis		nkonstantinou@iti.gr stalidis@iti.gr Michalis.Lazaridis@iti.gr

DOCUMENT HISTORY

Version #	Author name	Date	Changes
0.1	Nikolaos Konstantinou	01-03-2019	Starting version
0.2	Panagiotis Stalidis		Second version after corrections
0.3	Panagiotis Stalidis Michalis Lazaridis	20-03-2019	First complete draft, integrated with references
1.0	Theodoros Semertzidis	...	Reviewed and approved version

DOCUMENT DATA

Keywords	<i>Chatbot, Conversation agent, rule-based, deep learning, AI</i>
Editor Address data	Name: Theodoros Semertzidis Partner: CERTH Address: 6th km Charilaou Thermi rd, Thessaloniki, Greece Phone: +302310464160 (ext. 130) Email: theosem@iti.gr
Delivery Date	31-07-2020
Peer Review	<i>Laura, Luigi, UNINETTUNO Antonio Salvador, Filograna, ENG Pau, Pamplona, UAB</i>

EXECUTIVE SUMMARY

This report documents the work done thus far regarding the design and development of the Digital Companion, a hybrid chatbot that will be able to reply to common questions from both migrants and authorities. It describes the development of a rule-based chatbot as a first responder to frequent problems faced by migrants. A hand-off component handles routing of conversations to community members, volunteers or successfully integrated migrants when the chatbot can't manage to help with a situation. A guide is presented for the fast and effective addition of new conversation rules or the update of existing conversation trees through a graphical user interface. Additionally, it describes state-of-the-art methodologies used to create a modern conversational chatbot using machine learning technologies, to provide migrants practice in day to day life conversations ultimately leading to better inclusion in local societies and improved life quality of the migrants.

TABLE OF CONTENTS

Document Info	2
Authors	2
Document History	2
Document Data	2
Executive Summary	3
Table of Contents	4
Index of Tables	5
Index of Figures	5
Abstract	6
1. Introduction	6
2. Rule Based Chatbot	7
2.1. Implementation	8
2.1.1. Adding new rules to the Chatbot	8
3. Open Domain Dialogue Systems	13
3.1. Literature Review and Related Approaches	13
3.1.1. From Recurrent Neural Networks to Transformers	14
3.1.2. Transformer-Based Models for Conversational Agents	15
3.1.3. Focus on Conversational Skills of Dialogue Systems	17
3.1.4. Decoding Methods	18
3.1.5. Toolkits and Libraries for Dialogue Systems	19
3.2. A View to Transformer Architecture	20
3.3. The proposed Recommender Social Companion	21
3.4. Proposed Implementation for the Open-Domain Chatbot	23
3.4.1. Approach and methodology	23
3.4.2. Dataset for the Dialogue Task	24
3.4.3. Input data Representation	25
3.4.4. Fine-Tuning on Dialogue	26
3.4.5. Generation Method for the Chatbot	27
3.5. Requirements of a Multi-Language Conversational Agent	28
4. Conclusion	28
5. References	29

INDEX OF TABLES

Table 1. Average time in milliseconds to predict the next dialog utterance from N possible candidates	17
Table 2. Main results for comparing all decoding methods with selected parameters of each method (<i>Holtzman et al., 2019</i>).	19

Table 1

INDEX OF FIGURES

Figure 1 Full example of the conversation tree that was created for the health scenario of the pilots.....	8
Figure 2. Create a new blank diagram.	9
Figure 3. Add a start node	9
Figure 4. Change the color and add a label	10
Figure 5. Add properties by clicking "edit data..."	10
Figure 6. Add the property name	10
Figure 7. Add the caption to display for "english"	11
Figure 8. Add captions for the other languages and an "image" for the pictogram.....	11
Figure 9. Add two possible outcomes for this state: the user wants help with something and the user continues the conversation	11
Figure 10. By dragging from the arrow to a node, a connecting edge can be created	12
Figure 11. The same as with nodes, properties can be added using the "Edit Data..." option	12
Figure 12. Add responses for English, Spanish and Greek	12
Figure 13. Iterative loops can be created by dragging the edge to the originating node.....	13
Figure 14. Interactive sensibleness vs perplexity (Adiwardana et al., 2020)	16
Figure 15. . Interactive specificity vs perplexity (Adiwardana et al., 2020)	16
Figure 16. The transformer model-architecture from (Vaswani et. al.2017)	21
Figure 17. A Recommendation System	22
Figure 18. The Transformer architecture (Radford et al., 2018).....	24
Figure 19. Example dialog from the PERSONA-CHAT dataset. Person 1 is given their own persona (top left) at the beginning of the chat, but does not know the persona of Person 2, and vice-versa. They have to get to know each other during the conversation	25
Figure 20. Input representation for the dialogue task. Each token embedding is the sum of a word embedding, a dialog state embedding and a positional embedding.....	26
Figure 21. . Multi-task training objective - the model is provided with two heads for language modeling prediction (orange) and next-sentence classification (blue).....	27

ABSTRACT

This report summarizes the progress made to date in the design and development of Digital Companion for the purposes of REBUILD. Section 2 describes a rule-based chatbot which will be able to answer common questions from both migrants and authorities. This type of chatbot is designed to follow strict rules and will be used as the first to respond to potential critical problems that migrants may face. If a situation can not be addressed by the chatbot, the communication is redirected to community members, volunteers or successfully integrated migrants. One of the main issues the Rule-based chatbot had to address was the multi-language or even the illiteracy of the user base that prevented the creation of a single language chatbot. Additionally, section 3 analyzes the work done so far for the development of an open-domain multi-language chatbot and summarizes the literature reviews and related state-of-the-art approaches. The role of the open-domain chatbot is different from the rule-based. The open-domain chatbot will not be used to address critical issues, nor does it correspond to people who don't know the language, but will be able to mimic daily human-to-human conversations and to provide migrants with faster language awareness. This section outlines the proposed implementation and the requirements of such a multi-language open-domain conversational agent.

1. INTRODUCTION

A chatbot is a conversational agent capable of answering user queries in the form of text, speech, or via a graphical user interface. In simple words, a chatbot is a software application that can chat with a user on any topic. Chatbots can be broadly categorized into two types: rule-based chatbots, which are designed to perform specific tasks such as answer queries related to train reservation or medical appointments and open-domain chatbots which can have open-ended discussions with the users.

For the purposes of the REBUILD project, a chatbot is to act as a digital companion for the migrant. The digital companion has the role of answering common questions that are posed by either migrants or LSP's. Therefore, it is only natural that a rule based chatbot is developed. Rule-based chatbots, also referred to as decision-tree bots, use a series of defined rules as the basis for the types of problems the chatbot is familiar with and can deliver solutions for. Like a flowchart, rule-based chatbots map out conversations. They do this in anticipation of what a user might ask, and how the chatbot should respond. A key requirement that the REBUILD chatbot had to address was the multilinguality or even illiteracy of the user base that was holding a barrier to create a single language, written communication, chatbot.

On the other hand, open-domain chatbots use machine learning to understand the context and intent of a question before formulating a response. These chatbots generate their own answers to more complicated questions using natural-language responses. The more you use and train these bots, the more they learn and the better they operate with the user.

Rule-based chatbots are pretty straight forward as compared to learning-based chatbots. There are a specific set of rules. If the user query matches any rule, the answer to the query is generated, otherwise the user is notified that the answer to the user query doesn't exist. While rule-based bots have a less flexible conversational flow, these guard rails are also an advantage. You can better guarantee the experience they will deliver, whereas chatbots that rely on machine learning are a bit less predictable. Some of the advantages of a rule-based chatbot are that they are generally faster to train, are highly accountable and secure, can include interactive elements and media and can streamline the handover to a human agent, all very important requirements for the REBUILD project.

However, on the downside, rule based chatbots do not scale well. They can use very simple or complicated rules, they can't however answer any questions outside of the defined rules. These chatbots do not learn through interactions, they only perform and work with the scenarios you train them for. To add more responses, you have to define new rules.

2. RULE BASED CHATBOT

A rule based chatbot is algorithmically very simple: receive a message, check the set of rules to determine the state in which the conversation is, retrieve the answer for the current state and respond by sending a message back.

In the REBUILD platform, receiving and sending messages are handled by a communication layer developed for the platform. This communication layer provides endpoints for sending and retrieving messages. The messages are provided in JSON objects using "text" and "attachments" for the field to include the message. The "text" field is used for the text part of the message and is a string type. The "attachments" field is a list of URLs where media files, related to the message, can be downloaded from. These media can include pictures or videos sent by the user to ask a question, or images and videos sent back from the representative to answer a question visually. Additionally, since the REBUILD project is taking special care to allow communication even for illiterate migrants, attachments are also used to deliver pictograms along with the text part of the message.

As a first setup, a major concern for the chatbot, is to be able to easily identify the state that a conversation has reached in order to provide answers in a consistent way and avoid the frustration that a user might experience when a message is not understood by the system. A key decision in that direction is to avoid free text input from the user. For that reason, a number of possible messages has been defined for each of the states of conversation for the user to pick from. These comebacks from the user can be easily identified by the system and provide a stable transition from one state to the other. A number of special states have been defined for entering and leaving the conversation tree.

The first such state is the "START_NODE" state. This is the entering point to the conversation tree and is the first response of the chatbot in every new conversation.

On the other end, the "END_NODE" state defines when a conversation has finished. This allows the platform to clear any temporary resources assigned to the conversation. Any new messages from the same user after this state, trigger a new conversation.

At any point in the conversation the user must be able to declare that there is a communication problem and that in order to proceed a representative must be engaged. This functionality is provided by the "LIVE_CHAT" state. When a conversation is passed to the live representative, the chatbot is notified to transition to this state. When the live representative finishes her communication, the conversation can either be finished or resumed by the chatbot by a notification from the representative of what the chatbot's next state should be. During the conversation with the representative, all messages are ignored from the chatbot.

A final special case is if the user wants to engage in a social conversation. In this case that chatbot enters a special state "SOCIAL" where the conversation is passed to an open-ended chatbot handled by a different service.

In Figure 1, the conversation tree that was created for the *health scenario* can be found. In this example the "START_NODE", labeled as START is a green ellipse so that macroscopically the viewer of the graph can easily detect the beginning of the conversation. Respectively, the red ellipses are used for the "END_NODE" so that the end of the conversation is also easily detectable. Forwarding the conversation to a live representative is marked

with an orange ellipse, as is forwarding the conversation to the social bot. The blue boxes represent the questions posed by the chatbot while the white wavy boxes represent the possible answers of the user.

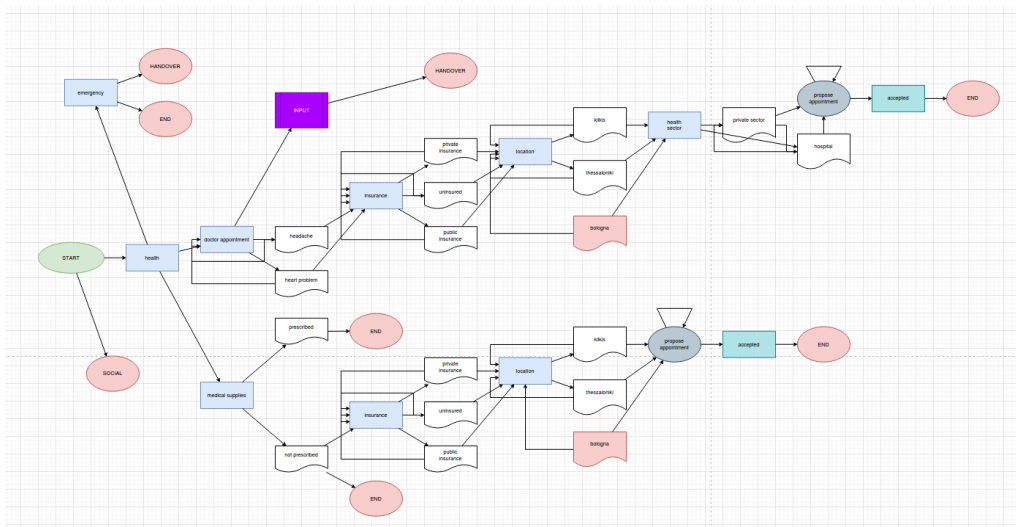


Figure 1 Full example of the conversation tree that was created for the health scenario of the pilots.

2.1. IMPLEMENTATION

Implementation wise, the chatbot was developed using *python* programming language. For communicating with the user interface and the other services of the platform, an endpoint was developed using the *flask* micro web services framework. The service listens for new messages to the chatbot and when a new message comes, it extracts the user that sent it and the contents of the message. Inside the message content, a payload object is enveloped so that the context to the previous state is passed to the chatbot.

Depending on the context, the appropriate state is selected. The various states of the conversation tree are stored in a graph object, created, maintained and traversed using the *networkx* library. This facilitates an ease of use since the conversation state can be identified by the graph's node's name as payload. After the selected node is identified, all the children nodes are retrieved to detect the next state of the conversation. The selected next state is sent back to the user in the form of an answer message.

Additional advantages in the use of *networkx* are that multiple trees can easily be merged into a single conversation and that it allows to dynamically edit the conversation tree by adding, removing or updating the graph.

The *networkx* graph can be defined in many different ways but in all cases the rules have to be written in a machine readable language and added to the system. One such language is XML (eXtensible Markup Language) which is a software and hardware independent language for storing and transporting data, recommended by W3C. Besides being software independent, key advantages of the XML language is that it has no predefined tags and as the name suggests it is extensible to add new tags at any point.

A helper function for reading an XML file and creating a new graph was also implemented. In the case where the XML is not syntactically correct, the function fails and the user is notified that the file had errors.

2.1.1. ADDING NEW RULES TO THE CHATBOT

Providing an even easier user experience, we selected the draw.io [ref] web application for the users to have a user interface for creating and updating conversations trees. The main features for selecting draw.io as a user interface is the fact that it is free to use, it allows sharing graphs between users in multiple formats such as google drive and it has the ability to export the created graph in XML.

Draw.io allows for adding any number of nodes and edges with predefined shapes. While the predefined shapes carry connotations to the user, for our purposes these connotations are disregarded during the translation to a conversation tree. The only distinction made is if the object is a node or an edge. Node objects are translated to answers from the bot while edges are translated to possible answers from the user. Additionally, each node or edge can have any number of properties assigned to it. We use this feature in order to pass information to the conversation tree regarding the text and/or image that should be displayed to the user during the traversal of the tree i.e. if the user has selected the English language, the system will search for the property "English" in order to extract the text caption to display.

A number of special nodes have to be defined for some specific cases. Such nodes are:

1. the first node of the conversation tree; which has to exist, and the node's label has to be set to "START"
2. any number of nodes to define that the conversation has finished; for these nodes the label has to be set to "END"
3. any number of nodes to define that the conversation should be forwarded to a live representative; this handover procedure is activated for nodes with the label "HANDOVER"

An example usage of draw.io to create the basic blocks of a conversation tree is:

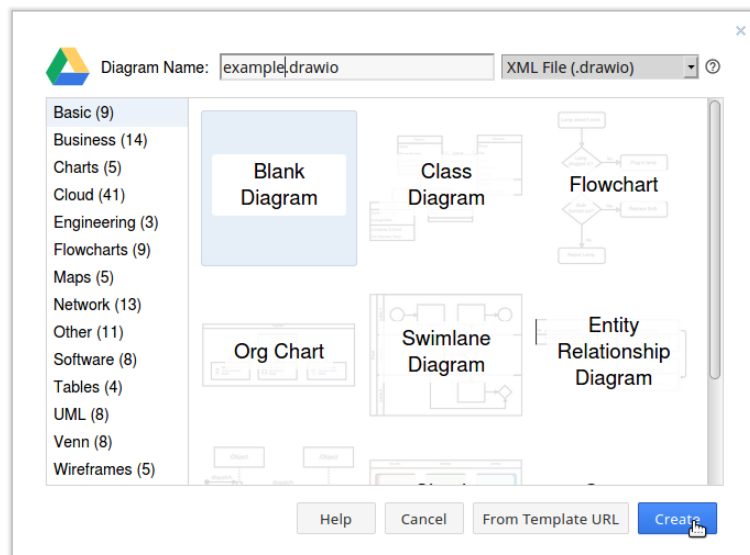


Figure 2. Create a new blank diagram.

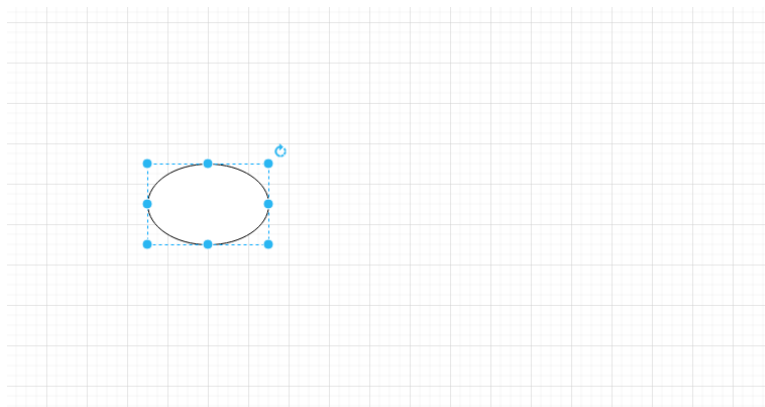


Figure 3. Add a start node

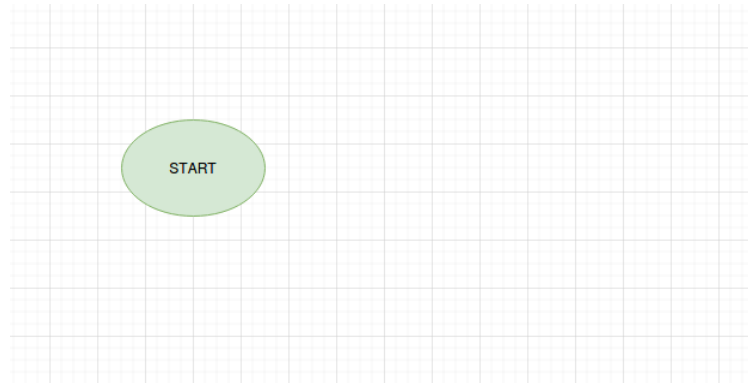


Figure 4. Change the color and add a label

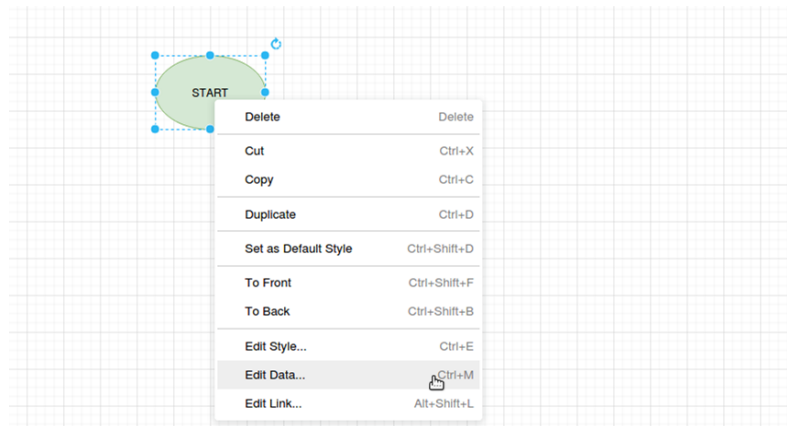


Figure 5. Add properties by clicking "edit data..."

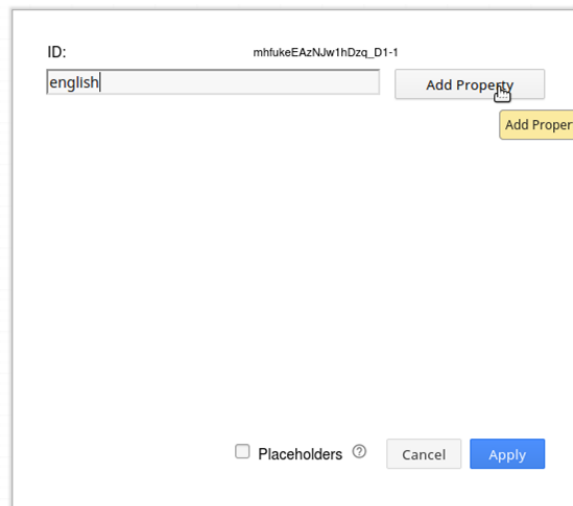



Figure 6. Add the property name



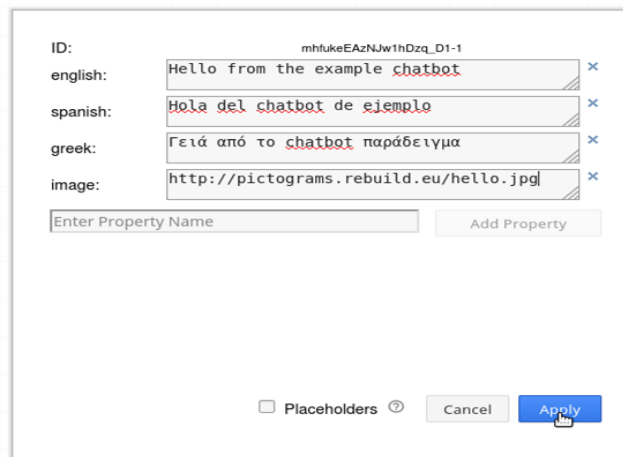
ID: mhfukeEAzNjw1hDzq_D1-1

english: Hello from the example chatbot

Enter Property Name Add Property

Placeholders ? Cancel Apply

Figure 7. Add the caption to display for "english"



ID: mhfukeEAzNjw1hDzq_D1-1

english: Hello from the example chatbot

spanish: Hola del chatbot de ejemplo

greek: Γειά από το chatbot παράδειγμα

image: http://pictograms.rebuild.eu/hello.jpg

Enter Property Name Add Property

Placeholders ? Cancel Apply

Figure 8. Add captions for the other languages and an "image" for the pictogram

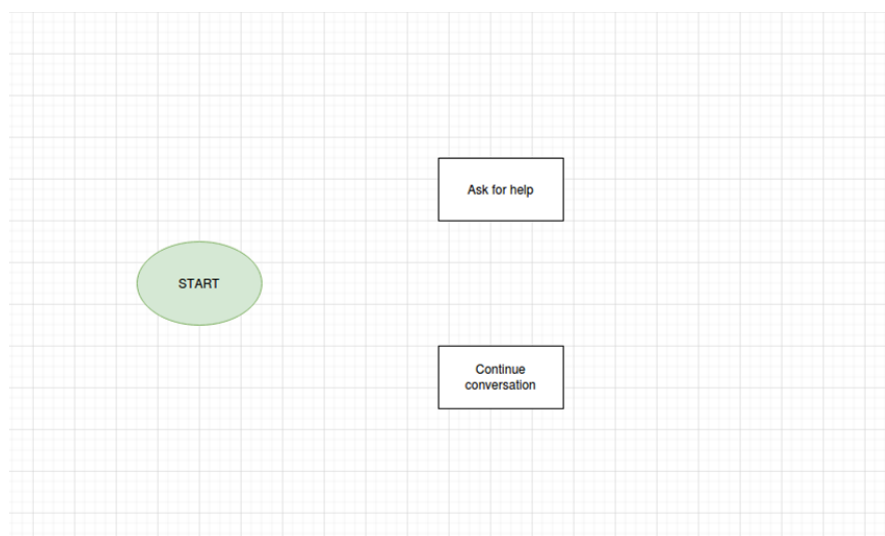


Figure 9. Add two possible outcomes for this state: the user wants help with something and the user continues the conversation

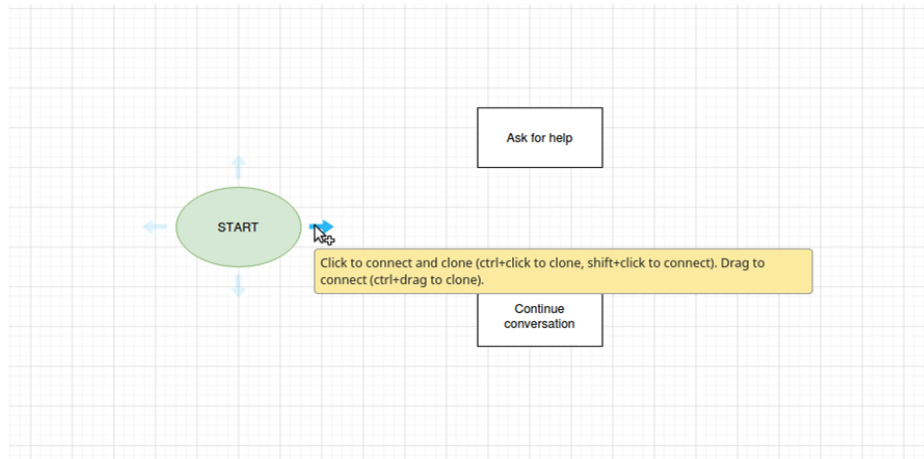


Figure 10. By dragging from the arrow to a node, a connecting edge can be created

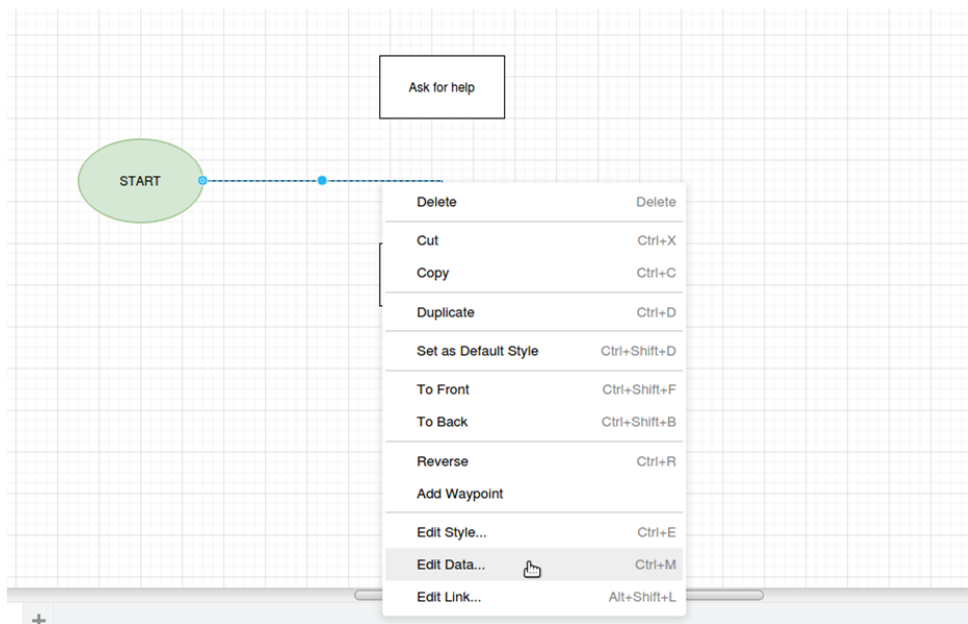


Figure 11. The same as with nodes, properties can be added using the "Edit Data..." option

ID: 47ZghVY-0koxHLrIXn3p-1

english: x

spanish: x

greek: x

Placeholders ⓘ

Figure 12. Add responses for English, Spanish and Greek

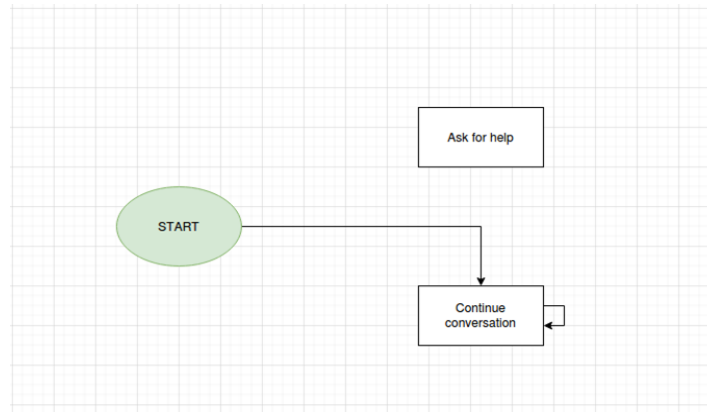


Figure 13. Iterative loops can be created by dragging the edge to the originating node

3. OPEN DOMAIN DIALOGUE SYSTEMS

For the purposes of REBUILD, a rule-based chatbot is developed that will provide significant guidance to migrants for daily problems and will be able to respond to common questions from both migrants and authorities in real time. In addition, except for the rule-based chatbot, an open-domain conversational agent that focuses on the social part is implemented. Rule-based chatbots are useful when we want to achieve a specific goal, such as hotel room reservations, ticket purchases, etc. Open-domain dialog systems (chatbots), on the other hand, focus more on interacting with people on a deeper level, and they do so by imitating human-to-human conversations. The role of the open-domain chatbot is to provide migrants practice in daily life conversations, that will ultimately lead to faster language awareness and to better adaptation in local societies.

3.1. LITERATURE REVIEW AND RELATED APPROACHES

Open-domain dialogue-systems are an interesting field of interactive Natural Language Processing (NLP) systems and they are useful in a wide range of applications, ranging from technical support services to entertainment. Building intelligent open-domain dialog systems that can converse with humans coherently and engagingly has been a long-standing goal of artificial intelligence. However, building intelligent conversational agents remains an unsolved problem in artificial intelligence research. A first interesting attempt at implementation of dialogue systems is with Recurrent Neural Networks (RNN) based models. On the one hand, neural dialogue models that use RNN, despite their success, still suffer from a lack of relevance, diversity, and in many cases coherence in the generated responses. These issues can be attributed to reasons including short-range model architectures that capture limited temporal dependencies, the concave entropy profile of dialogue datasets resulting in short and generic responses, and the out-of-vocabulary problem leading to generation of a large number of out-of-vocabulary <UNK> tokens.

On the other hand, with the recent success of the attention mechanism, Transformer-Based models demonstrate excellent ability to capture long-range structures in language modeling tasks. The Transformer architecture allows the creation of NLP models trained on huge datasets. The weights learned by such massive pre-trained models can later be reused for specific tasks (as dialogue) by fine-tuning them to the specific dataset. In addition, except for the architecture of the model, there are several modelling strategies that can provide important improvements

and solutions on the issues above, such as the kind of dataset for the training procedure or the decoding algorithm for the generation task. Good conversation requires a number of skills as: providing engaging talking points, displaying knowledge, empathy and personality appropriately, while maintaining a consistent persona. Large improvements can be made by fine-tuning on a dataset which targets on those aspects by providing training data and initial conversational context (personas and topics). Additionally, the decoding algorithm is a part of critical importance in the development of a Conversational Agent, as it is responsible to transfer the “knowledge” of the trained model to a comprehensible language for human beings.

3.1.1. From Recurrent Neural Networks to Transformers

Recurrent Neural Networks that are capable of generating meaningful responses in some dialogue tasks are demonstrated by (Vinyals and Le 2015). However, further research in the capabilities of these neural network architectures and developments (Serban et al., 2016; Miao et al., 2015; Sordani et al., 2015; Serban et al., 2017; Li et al., 2016; Li et al., 2017) indicate that they are limited, which makes the communication between people and models a rather unsatisfying experience for human beings. The main issues with these architectures can be summarized as:

- The inconsistent outputs and the lack of a consistent personality (Li and Jurafsky, 2016).
- The absence of a long-term memory, as these models have difficulties to take into account more than the last dialogue sentence.
- A tendency to produce consensual and generic responses (e.g. I don't know) which are vague and not engaging for humans (Li et al., 2016).

The Transformers era originally started after 2017 by the work of (Vaswani et al. 2017). This work demonstrates the superiority of these models over RNNs on translation tasks, but their superiority was quickly extended to almost all the tasks RNNs were State-of-the-Art at that time. RNN's sequential nature makes them unsuitable for increasingly larger datasets and exponentially slower to be trained on larger datasets. An important advantage of Transformer over its RNN counterpart is its non sequential attention model. In the Transformer architecture proposed by Vaswani, RNN cells are replaced with self-attention and pointwise fully connected layers, which are highly parallelizable and thus cheaper to compute. While the RNNs iterate over each element of the input sequence one-by-one, and carry an “updatable-state” between each hop, using transformer-based models has the advantage of looking at every position in the sequence, at the same time in one operation.

Recent advances in large-scale pre-training using transformer-based architectures (Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2019) achieve great empirical success. While some of the above issues (such as the inconsistent personality) can be alleviated by modelling strategies specifically designed to boost information content, a transformer-based architecture like Generative Pretrained Transformer-2 (GPT-2) (Radford et al., 2018), which uses a multi-layer self-attentive mechanism to allow fully-connected cross-attention to the full context in a computationally efficient manner, is a natural choice for exploring a more general solution. Transformer models allow long-term dependency information to be better preserved across time (Radford et al., 2018), thereby the content consistency is improved. They also have higher model capacity due to their deep structure (up to 48 layers in GPT-2) and they are more effective in leveraging large-scale datasets (more than 100 million training instances) than RNN-based approaches.

3.1.2. Transformer-Based Models for Conversational Agents

Building open-domain chatbots is a challenging area for machine learning research. Our interaction with machines is crucial for any future application of intelligent agents. There are various ways for a model to determine what to say next in a conversation, though these methods can be distilled into two main approaches: generative models, which generate a sequence of text, and retrieval/ranking models, which rank candidates among a fixed set and select the optimal next utterance for a model. Multiple works have recently used transformer architectures in dialogue modeling.

Two recent transformer-based models that use GPT-2 as base and they are trained for the task of dialogue, are DialoGPT (Zhang *et al.*, 2019) and DLGnet (Olabiyi and Mueller, 2019). The GPT-2 transformer model adopts the generic transformer language model (Vaswani *et al.*, 2017) and leverages a stack of masked multi-head self-attention layers to train on massive text data.

DialoGPT (Dialogue Generative Pre-trained Transformer)(Zhang *et al.*, 2020) inherits a GPT-2 architecture and it is used with three different sizes of total parameters: 117M, 345M and 762M. It uses a 12-to-48 layer decoder with layer normalization, an initialization scheme that accounts for model depth that is modified, and Byte Pair Encodings (BPE)(Sennrich *et al.*, 2016) for the tokenizer which solves the out-of-vocabulary problem. A simple form of data compression as BPE can be a practical middle ground between character and word level language modeling which effectively interpolates between word level inputs for frequent symbol sequences and character level inputs for infrequent symbol sequences. Unlike GPT-2, however, DialoGPT is trained on large-scale dialogue pairs/sessions extracted from Reddit discussion chains. DialoGPT extends GPT-2 to address the challenges of conversational neural response generation and attains a performance close to human, both in terms of automatic and human evaluation in single-turn dialogue settings.

A similar model DLGnet (Olabiyi and Mueller, 2019), an extension of autoregressive transformer model that uses GPT-2 as base, is trained on the open-domain Movie Triples dataset and the closed-domain Ubuntu Dialogue dataset and achieving significant improvements over state-of-the-art multi-turn dialogue models. The model analysis shows that the performance improvement is mostly due to the combination of (i) the long-range transformer architecture with (ii) the injection of random informative paddings exploiting the large maximum input sequence. Other contributing factors include the joint modeling of dialogue context and response, and the 100% tokenization coverage from the byte pair encoding.

The ConvAI2 challenge at NeurIPS (Dinan *et al.*, 2019) focuses on personalized conversations, where a variant of PERSONA-CHAT dataset is used. The best models in the competition were variants of the generative Transformer architecture. In particular, (Wolf *et al.* 2019) uses a generative transformer model with 12 layers (768 dimensional states and 12 attention heads) which is pre-trained via the method of (Radford *et al.*, 2018) using the Books Corpus dataset (7,000 unique unpublished books). Then the model is fine-tuned in the dialogue task with the PERSONA-CHAT dataset. A multi-task loss (consisting of next token and next sentence prediction) is optimized by (Wolf *et al.* 2019), resulting in the best perplexities and F1 scores in the competition. The repository of *Hugging-face Conv-AI transfer learning* can be used to reproduce the results of Hugging-Face's participation in NeurIPS 2018 dialog competition ConvAI2 which was state-of-the-art on the automatic metrics.

Another particular large-scale model of note is Meena (Adiwardana *et al.*, 2020), a 2.6B parameter generative Transformer-based model trained on 341 GB of text (40B words), that has shown to be superior to variants of DialoGPT. The main architecture of Meena is a seq2seq model (Sutskever *et al.*, 2014; Bahdanau *et al.*, 2015) with the Evolved Transformer (So *et al.*, 2019). This model is trained on multi-turn conversations where the input sequence is all turns of the context and the output sequence is the response. Additionally, this work aims at finding a representative automatic metric that corresponds with human evaluation which always has been an important goal of open-domain conversational modeling. The Sensibleness and Specificity Average (SSA) metric is proposed, which captures key elements of a human-like multi-turn conversation. Their experiments show strong correlation between perplexity and SSA as is demonstrated in Figures 14 and 15.

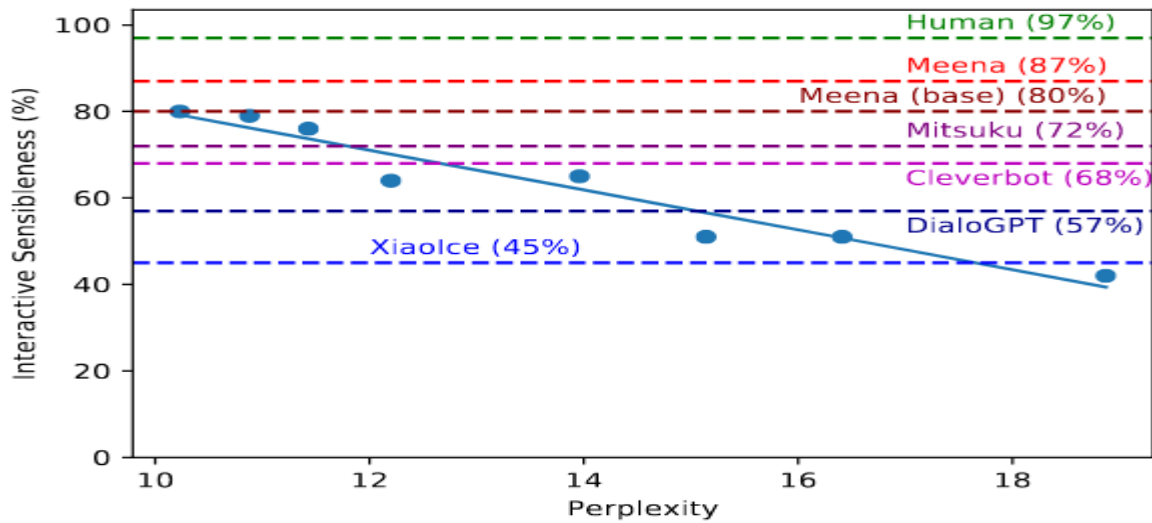


Figure 14. Interactive sensibleness vs perplexity (Adiwardana et al., 2020)

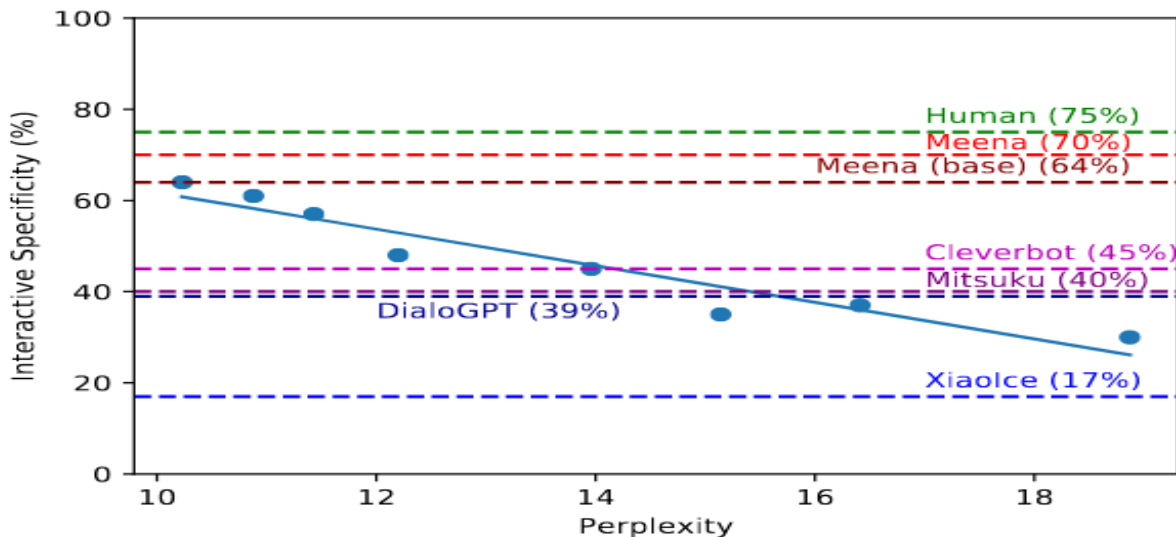


Figure 15. . Interactive specificity vs perplexity (Adiwardana et al., 2020)

On the other hand, given a dialogue history (context) as input, retrieval/ranking systems select the next dialogue utterance by scoring a large set of candidate responses and outputting the highest scoring one. General, all possible training set responses are used as the candidate set.

In the work of (Humeau et al., 2019) a comparison among three different architectures of large retrieval transformer pretrained models, Bi-encoders, Cross-encoders, and Poly-encoders is performed. They experiment in two different pre-training schemes. In the first the weights of the models are initialized from those of the transformer model proposed in (Devlin et al., 2018) that was trained on a dataset combining sentences from Wikipedia and Toronto Books Corpus. While in the second pre-training scheme the models have been trained on a dataset of 800 million comments derived from the online platform Reddit. For the fine-tuning procedure ConvAI2 and DSTC7 datasets are used. The fine-tuning procedure involves the training of the models in order to predict the next sentence in a dialogue given the dialogue context/history. In addition, they perform speed experiments for these models (Table 1). Specifically, they predict the next utterance for 100 dialogue examples in the ConvAI2 validation set, where the model has access to N candidates from the train set.

Their results show that the pre-training scheme on Reddit comments outperforms the work in (Devlin et al., 2018) for all three model architectures. Though the Cross-encoder architecture yields the highest results in all metrics, it is prohibitively slow (Table 1). Poly-encoder model according to their experiments achieves better results than the Bi-encoder and obtains an accuracy close to the Cross-Encoder while also performing at a reasonable speed.

Model	Scoring time (ms)			
	CPU		GPU	
Candidates	1k	100k	1k	100k
Bi-encoder	115	160	19	22
Poly-encoder 16	119	551	17	37
Poly-encoder 64	124	570	17	39
Poly-encoder 360	160	837	17	45
Cross-encoder	21692	-	2655	-

Table 1. Average time in milliseconds to predict the next dialog utterance from N possible candidates

3.1.3. Focus on Conversational Skills of Dialogue Systems

Reviews from recent works on neural approaches are presented by (Huang and Zhu 2020), that are devoted to addressing three challenges in developing chatbot systems: semantics, consistency, and interactiveness. One strategy to approach these challenges is by providing to the model an appropriate dataset, which targets personality, engagingness, knowledge, and empathy.

The work of (Zhang et al., 2018) focuses on improving the issue of inconsistent outputs and the lack of a consistent personality in a conversational agent. In this paper, the PERSONA-CHAT dataset is presented. The dataset consists of crowd-sourced dialogues, where each participant plays the part of an assigned persona and each (crowd-sourced) persona has a word-distinct paraphrase. PERSONA-CHAT is tested on two classes of models for next utterance prediction: ranking models and generative models. The results from this work demonstrate that both classes of models that have access to their own personas in addition to the state of the dialogue, are scored as more consistent by annotators who evaluate the fluency, the engagingness and the consistency of each model. In addition, it is indicated that models trained on PERSONA-CHAT (with or without personas) are more engaging than models trained on dialogue from other resources (movies, Twitter).

The Empathetic Dialogue (ED) dataset is constructed by (Rashkin et al., 2019), which consists of 50k utterances of crowd-worker conversations grounded in an emotional situation. In each dialogue, one speaker describes a personal situation and the other plays a "listener" role, displaying empathy during the discussion. It has been shown that fine-tuning models on this dataset helps them display more empathy in human evaluations. A similar work is proposed in (Dinan et al., 2019) in which the Wizard of Wikipedia (WoW) dataset is presented. This task involves discussing a given topic in depth, where the goal is to both engage the partner and display expert knowledge. The dataset consists of 194k utterances over 1250 topics, where each conversation begins with a randomly chosen topic. It is indicated in their human evaluation procedure, that for the generative models the human engagingness ratings are significantly improved by the use of such knowledge.

The goal of (Smith et al. 2020) is to create an open-domain conversational agent that can display many conversational skills. It aims to blend the previous three tasks to combine the skills from them (engaging personality from PERSONA-CHAT, empathy from ED, and knowledge from WoW) seamlessly during dialogue. The proposed work of (Smith et al., 2020) demonstrates several ways to leverage previous attempts that focus on individual conversational skills, by combining trained single-skill models in a two-stage way, by re-using the datasets for simultaneous multi-task training, and by fine-tuning on the overall blended task. They introduce Blended Skill Talk (BST), a new English-language dataset blending the three conversation skills in balanced proportions (demonstrating knowledge, empathy, or ability to talk about oneself). The base architecture used

throughout this work is the 256-million parameter poly-encoder proposed in (Humeau et al., 2019). The poly-encoder is first pre-trained on pushshift.io Reddit dataset and then is fine tuned on the BST dataset. Their experiments indicate that multi-tasking over several tasks that focus on particular capabilities, results in better blended conversation performance compared to models trained on a single skill.

Another work where the BST dataset is tested, is on (Stephen Roller et al., 2020), where three types of architectures are used: retrieval, generative, and retrieve-and-refine models. All three use Transformers as a base. In this kind of work variants of these recipes with 90M, 2.7B and 9.4B parameter models are demonstrated. The results from this work have shown that:

(i) Large improvements can be made by fine-tuning on data that emphasizes desirable conversational skills using the recently introduced BST (Smith et al., 2020) set-up. Dialogue systems achieve large gains by using the BST dataset, which targets personality, engagingness, knowledge, and empathy by providing training data and initial conversational context (personas and topics). Small models that use this dataset can match or outperform larger models that do not.

(ii) The length of the bot's utterances are crucial to human judgments of quality – too short and the responses are seen as dull or showing a lack of interest, too long and the bot appears to waffle. It indicates, contrary to previous work which reports that beam search is inferior to sampling (Holtzman et al., 2019; Adiwardana et al., 2020), that careful choice of search hyper-parameters can give strong results by controlling trade-offs. In particular, constraining the minimum beam length gives a crucial control of the dull versus spicy spectrum of responses.

A different kind of approach is presented in the work of (He Bai et al., 2020), where the importance of adding special tokens such as end-of-paragraph (EOP) and end-of-sequence (EOS) are studied. Their experiments based on Transformer-based pretrained language models (LMs), showed that these models can generate a text with higher quality when paragraph and sentence information are provided. The basic assumption of auto-regressive generation is that the probability of a word sequence equals the product of conditional word probability (W_0 refers to the given context):

$$p(w_{1:T}|W_0) = \prod_{t=1}^T P(w_t|w_{1:t-1}, W_0) \quad (1)$$

The generated sequence length, T is usually determined by the time t generating the EOS (end-of-sequence) token:

$$p(w_T|w_{1:T-1}, W_0) = P(EOS|w_{1:t-1}, W_0) \quad (2)$$

3.1.4. Decoding Methods

The choice of decoding algorithm is of critical importance in the development of a Chatbot. Two models with the same perplexity scores but different decoding algorithms can give vastly different results in the communication with human beings. The two most common decoders for language generation are greedy-decoding and beam-search.

Greedy-decoding's method selects the most probable word from the model's vocabulary at each decoding time-step as the candidate for the output sequence. The problem with this approach is that once the output is chosen at any time-step t , it doesn't have the flexibility to go back and change its choice. Additionally, in practice it seems that greedy decoding strategy is prone to have grammatical errors in the generated text. It results in choosing the best choice at any time-step t but that might not necessarily give the best result when considering the full sentence to be grammatically correct and sensible.

Especially, over the last few years, beam-search has been the standard decoding algorithm for almost all language generation tasks including dialog (Ilija Kulikov et al., 2018). Several developments occurred in 2018-2019, at first

indicating that beam-search is strongly sensitive to the length of the outputs and best results could be obtained when the output length is predicted before decoding (*Kenton Murray et al., 2018; Yilin Yang et al., 2018*). While this makes sense for low-entropy tasks like translation, where the output sequence length can be roughly predicted from the input, it seems arbitrary for high-entropy tasks like dialog and story generation, where outputs of widely different lengths are usually equally valid.

In parallel with (*Kenton Murray et al., 2018; Yilin Yang et al., 2018*), two influential works (*Radford et al., 2018; Angela Fan et al., 2018*) on high-entropy generation tasks were proposed in which greedy/beam-search decoding are replaced by *sampling* from the next token distribution at each time step. These works use a variant of sampling called top-k sampling in which the decoder samples only from the top-k most-probable tokens (k is a hyper-parameter).

A last paper, in this recent trend of work, is the work recently published by (*Ari Holtzman et al., 2020*) which shows that the distributions of words in texts generated using beam-search and greedy decoding is different from the distributions of words in human-generated texts (Table 2). Clearly, beam-search and greedy decoding fail to reproduce some distributional aspects of human texts as it has also been noted in (*Jason Weston et al., 2018; Emily Dinan et al., 2019*) in the context of dialog systems.

Method	Perplexity	Self-BLEU4	Zipf Coefficient	Repetition %
Human	12.38	0.31	0.93	0.28
Greedy	1.50	0.50	1.00	73.66
Beam, b=16	1.48	0.44	0.94	28.94
Stochastic Beam, b=16	19.20	0.28	0.91	0.32
Pure Sampling	22.73	0.28	0.93	0.22
Sampling, $t=0.9$	10.25	0.35	0.96	0.66
Top- $k=40$	6.88	0.39	0.96	0.78
Top- $k=640$	13.82	0.32	0.96	0.28
Top- $k=40, t=0.7$	3.48	0.44	1.00	8.86
Nucleus $p=0.95$	13.13	0.32	0.95	0.36

Table 2. Main results for comparing all decoding methods with selected parameters of each method (*Holtzman et al., 2019*)

3.1.5. Toolkits and Libraries for Dialogue Systems

There are several open-sourced toolkits for large-scale pre-trained transformer models and conversational agents. An open-source library is *Transformers* (formerly known as *pytorch-transformers* and *pytorch-pretrained-bert*) which provides general-purpose transformer-based-architectures for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages. *Hugging-face Conv-AI transfer learning* repository (*Wolf et al. 2019*) contains the code for training conversational AI systems with transfer learning based on the *Transformers* library, which achieves the state-of-the-art performance on ConvAI2 dialogue competition. *DeepPavlov* (*Burtsev et al., 2018*) is a popular framework focusing on task-oriented dialogue. This public repository contains several demos and pre-trained models for question-answering and sentiment classification. *Icecaps* is a response generation toolkit with techniques such as grounding on personalities or external knowledge and multi-task training. *ParAI* (*Miller et al., 2017*) is another library for developing task-oriented dialogue systems. It contains pre-trained models for knowledge-grounded chatbot trained with crowd-sourced data.

3.2. A VIEW TO TRANSFORMER ARCHITECTURE

The model was first introduced by *Vaswani* in the paper titled (*Attention is all you need. 2017*). Transformer networks can be applied to many NLP tasks successfully. The architecture consists of an encoder and a decoder (Figure 16) that work non-recurrently, therefore, overcoming the sequentiality problem that RNNs suffer from. Transformers take advantage of variants of attention mechanism in many aspects of their networks to calculate various values.

A basic attention mechanism that is calculated through dot product has as input, a query q and a set of key-value ($k - v$) pairs and produces an output vector. The output is obtained by the weighted sum of values where the weight of each value is computed by an inner product of the query and the corresponding key. Queries and keys have the same dimensionality. A basic attention model can be expressed as follows:

$$A(q, k, v) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} u_i \quad (3)$$

Transformers take full advantage of various capabilities of attention. The input word vectors are the queries, keys, and values. Transformers introduce a new idea: Multi-headed self-Attention. Multi-head self-attention enables the words to interact with one another in more than one way as they do in a simple self-attention. Multi-head attention maps Q, K, V into lower dimensional spaces via W matrices and applies attention before it concatenates the output to a linear layer. This method allows the model to 'attend to' multiple parts of a sentence at the same time.

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^o \quad (4)$$

where:

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (5)$$

Each complete transformer block has a multi-head attention and a 2-layer feed-forward NN with Rectified Linear Unit (ReLU) activation function. Each of these also has a residual connection and a Layer Normalization. The input to the encoder are word representations in the form of byte-encodings as well as positional encodings allowing the same word to have a different representation at a different location in the sentence. At each block, repeated six times, the encoder uses the same $Q, K,$ and V from the previous layer. As the blocks are repeated, words start to pay attention to other words in meaningful ways, making connections.

Figure 16. The transformer model-architecture from (*Vaswani et. al.2017*)

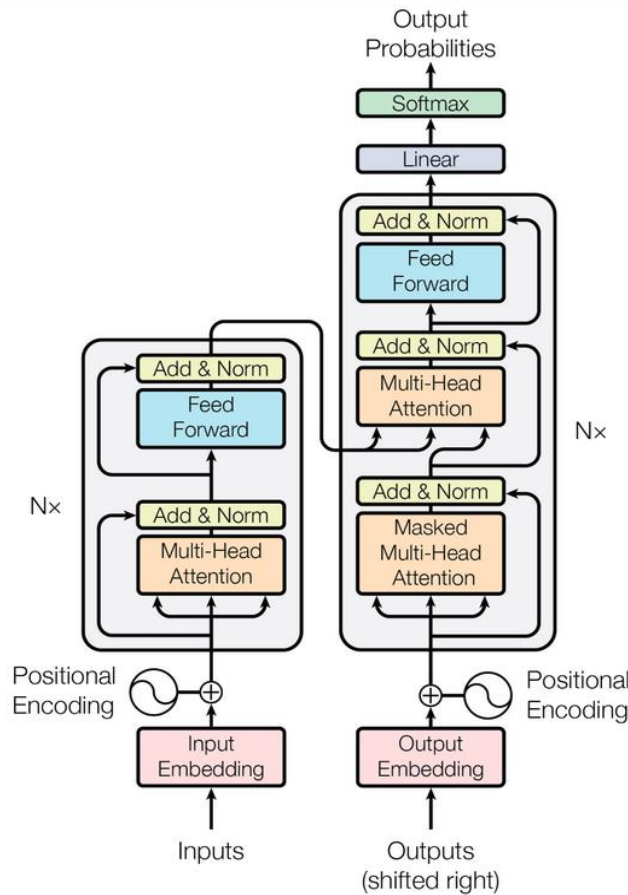


Figure 16. The transformer model-architecture from (Vaswani et. al.2017)

In the encoder, with the multi-head attention, the model is able to attend different parts of the input by calculating values and feeding them forward to the fully connected layer and repeating it as it goes deeper. This allows pushing information progressively along with the sequence in either direction to calculate values that are interesting for a given context. Then, the decoder implements self-attention on previously generated outputs. Self-attention, with the ability to represent a specific word based on the weighted combination of the entire neighboring words, brings many advantages, such as a constant path length between any two positions in the sequence, multiplicative interactions (gating), and easy concurrent operations.

3.3. THE PROPOSED RECOMMENDER SOCIAL COMPANION

The purposes of REBUILD require a Conversational Agent that will ultimately lead to better inclusion in local societies and improved quality of life for migrants. Except for the creation of a Conversational Agent that would certainly help migrants meet a new language, a recommendation system (Figure 17) in combination with the dialogue system would provide important guidance to migrants to meet the local societies.

A Chatbot is built on two main pillars:

- A. Dialogue system: The Chatbot is capable of conducting a written or verbal conversation with a user.
- B. Recommendation system: The Chatbot enhances the user experience by finding similarities among users and/or items and generating a list of items matched to end-user's preference.

The dialogue system can be the primary interface for user's interaction and a variety of recommendations such as local events or activities. Users could also interact with the dialogue interface to modify their preferences captured in the conversation or to interact with the results of recommendations. The aim of this recommendation system is to increase the level of user engagement by providing more flexible and personalized guidance by enabling interactions with a natural language.

While other platforms, such as natural language search, take user's intents and preferences in a natural language and return the recommended items, building a recommendation system with a dialogue system requires the support of continuous interactions between a dialogue system and users. In conventional recommender systems, personalized recommendation is highly based on the previous actions of users, including searching or clicking. These actions can be regarded as feedback from users reflecting their interest. However, due to its implicit nature, such feedback can only reflect a part of the user's interest, which causes the recommendation to be inaccurate. A dialog between users and services is another source of information on user preferences. Users often provide more information about their preferences in such a dialog where they often ask for tips or recommendations. In this process, services can guide them to speak out about their interests in order to solve the problems of users and meet their requirements. Compared to implicit feedback, feedback from the dialog is more explicit and related to users preferences. A recommender dialog system can be considered as a combination of a recommender system and a dialog system. A dialog system should respond to user statements with informative natural language expressions, and a recommendation system should provide high-quality recommendations based on the content of the user's statements.

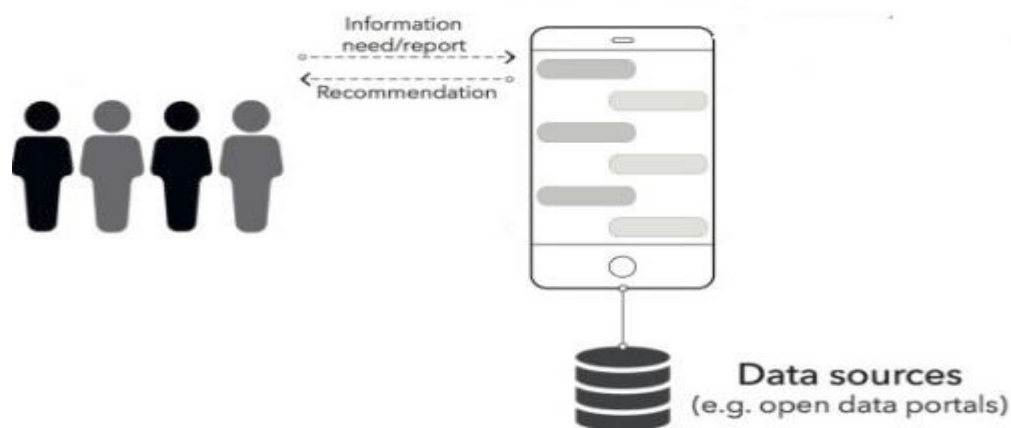


Figure 17. A Recommendation System

In brief, a recommender-dialog system should perform well in both tasks. An ideal recommender-dialog system is an end-to-end framework that can effectively integrate the two systems so that they can bring mutual benefits to each other. Information from the recommender system can provide vital information to maintain multi-turn dialog, while information from the dialog system that includes user preferences may improve the quality of recommendation. In addition, the incorporation of external knowledge can strengthen the connections between systems and improve their performances.

There are two main challenges to the recommendation of personalised suggestions. The first is to obtain the data sources that will be used for the matching with the preferences of the user. The second challenge is the personalisation itself: Building a system that can respond in real-time to the constraints that users may express through the chat application, while considering activities and locations that match their preferences.

3.4. PROPOSED IMPLEMENTATION FOR THE OPEN-DOMAIN CHATBOT

3.4.1. Approach and methodology

At the core of our approach is language modeling. Language modeling is usually framed as unsupervised distribution estimation from a set of examples (x_1, x_2, \dots, x_n) each one composed of variable length sequences of symbols (s_1, s_2, \dots, s_n) . Since language has a natural sequential ordering, it is common to factorize the joint probabilities over symbols as the product of conditional probabilities (*Bengio et al., 2003*):

$$p(x) = \prod_{i=1}^n p(s_i | s_1, \dots, s_{i-1}) \quad (6)$$

This approach allows tractable sampling and estimation of $p(x)$ as well as any conditionals of the form $p(s_{n-k}, \dots, s_n | s_1, \dots, s_{n-k-1})$. In recent years, there have been significant improvements in the expressiveness of models that can compute these conditional probabilities, such as self-attention architectures like the Transformer (*Vaswani et al., 2017*).

For the development of the open-domain chatbot, a multi-layer Transformer model is proposed, based on the GPT-2 of (*Radford et al., 2018*) and the open source implementation of *Hugging-face Conv-AI transfer learning* repository (*Wolf et al. 2019*). In the literature this version of the Transformer is often referred to as a Transformer-decoder since it is similar to the decoder part of the original encoder-decoder Transformer of (*Vaswani et al., 2017*).

The GPT-2 is similar to the large Transformer model recently used in several works, leading to impressive results on several downstream NLP tasks (*Radford et al., 2018; Devlin et al., 2018*). It uses a 12-layer decoder-only transformer with masked self-attention heads (768 dimensional states and 12 attention heads). By masked attention, is meant that the Transformer uses constrained self-attention where every token can only attend to its left context. GPT-2 is a stacked decoder-Transformer, which inputs a sequence of tokens and applies position and token embeddings followed by several decoder layers. Each layer applies multi-head self-attention in combination with a feedforward network, layer normalization and residual connections (Figure 18). For the activation function, it uses the Gaussian Error Linear Unit (GELU) (*Hendrycks and Gimpel, 2018*). In the proposed format (*Radford et al., 2018; Devlin et al., 2018*) the model uses learned positional embeddings with supported sequence lengths up to 1024 tokens. The input sentences are pre-processed and tokenized using byte pair encoding (*Sennrich et al., 2016*).

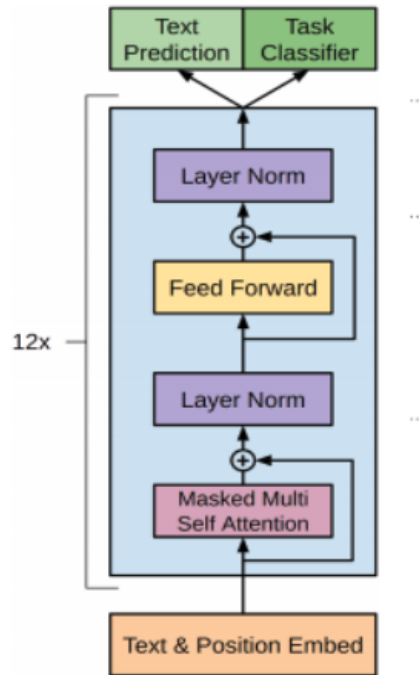


Figure 18. The Transformer architecture (Radford et al., 2018)

The methodology of the implementation can be described in two main phases:

- Start by pretraining a language model on a very large corpus of text to be able to generate long stretches of contiguous coherent text. In the implementation of (Wolf et al. 2019), the model uses the pre-trained model weights open-sourced by Radford. This model is pre-trained on the BooksCorpus dataset (Zhu et al., 2015) which contains over 7,000 unpublished books (about 800M words) from a variety of genres. However, a more appropriate pre-training scheme for dialogue tasks as recently used by DIALOGPT (Zhang et al., 2020) is proposed. DIALOGPT is trained on large-scale dialogue pairs/sessions extracted from Reddit discussion chains. This dataset comprises 147,116,725 dialogue instances, in total 1.8 billion words. The DIALOGPT's size corresponds to GPT-2 small, with 12 layers and 117M parameters (Radford et al., 2018).
- Fine-tune the pre-trained model to the PERSONA-CHAT dataset by using the implementation of (Wolf et al. 2019). More precisely, our model uses the open sourced-weights of the DIALOGPT and is fine-tuned to the PERSONA-CHAT dataset using an augmented input representation and a multi-task learning scheme, further described in following sections. The model finetunes quickly to the PERSONA-CHAT dataset and 5 epochs of training was sufficient. A batch size of 4 and accumulated gradients over 8 iterations are used, resulting in the effective batch size of 32. In addition, the Adam optimization scheme with a learning rate of 6e-5 is used.

3.4.2. Dataset for the Dialogue Task

For the task of dialogue and the fine-tuning procedure a dataset based on the PERSONA-CHAT dataset (Zhang et al., 2018) is used. It is a crowd-sourced dialogue dataset in which each speaker was asked to condition its expressions on a predefined profile comprising a few sentences defining a personality as illustrated on Figure 19. Paired workers were asked to chat naturally and to get to know each other during the conversation. The aim of this dataset is to facilitate more engaging and more personal chit-chat dialogue. A model trained on this kind of dataset can lead to produce more personal, specific, consistent and engaging responses than a persona-free

model, thus alleviating some of the common issues(as the inconsistent personality) in chit-chat models. PERSONA-CHAT is available in raw tokenized text format in Facebook's *ParlAI* library.

Persona 1	Persona 2
I like to ski My wife does not like me anymore I have went to Mexico 4 times this year I hate Mexican food I like to eat cheetos	I am an artist I have four children I recently got a cat I enjoy walking for exercise I love watching Game of Thrones

[PERSON 1:] Hi
 [PERSON 2:] Hello ! How are you today ?
 [PERSON 1:] I am good thank you , how are you.
 [PERSON 2:] Great, thanks ! My children and I were just about to watch Game of Thrones.
 [PERSON 1:] Nice ! How old are your children?
 [PERSON 2:] I have four that range in age from 10 to 21. You?
 [PERSON 1:] I do not have children at the moment.
 [PERSON 2:] That just means you get to keep all the popcorn for yourself.
 [PERSON 1:] And Cheetos at the moment!
 [PERSON 2:] Good choice. Do you watch Game of Thrones?
 [PERSON 1:] No, I do not have much time for TV.
 [PERSON 2:] I usually spend my time painting: but, I love the show.

Figure 19. Example dialog from the PERSONA-CHAT dataset. Person 1 is given their own persona (top left) at the beginning of the chat, but does not know the persona of Person 2, and vice-versa. They have to get to know each other during the conversation

For the goals of REBUILD, datasets with dialogues and in other languages such as Greek and Italian, are required in order to fine-tune the pre-trained model. Creation of a synthetic dataset with short (6-7 turns of dialogue) dialogues in both languages is proposed, where each dialogue is performed between two specific personalities containing 4-5 phrases (15 maximum words per phrase). The aim is to create personas that are natural and descriptive and contain typical topics of human interest that the speaker can bring up in conversation. Moreover, for REBUILD's specific purposes, the aim is the personas and the dialogues to have elements that correspond in each country's local culture.

3.4.3. Input data Representation

As mentioned in previous sections, byte pair encoding is used at the core of the input representation in GPT-2. Byte pair encoding is a compression procedure in which a list of subwords is calculated using the following algorithm:

- Split word to sequence of characters.
- Joining the highest frequency pattern.
- Keep doing the previous step until it reaches the predefined maximum number of iterations subwords.

For the task of dialogue, the model's input must be adapted to the PERSONA-CHAT dataset. In addition to personality sentences like the one of the PERSONA-CHAT dataset, a two-speaker environment is required for the input representation of the Chatbot. In order to generate an output sequence, the model will have to use two kinds of contexts in a dialog setting:

- one or several persona sentences,
- the history of the dialog with at least the last expression from the user.

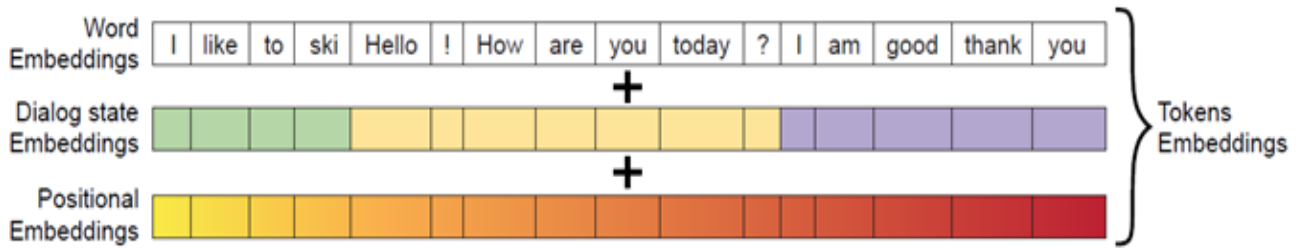


Figure 20. Input representation for the dialogue task. Each token embedding is the sum of a word embedding, a dialog state embedding and a positional embedding

More precisely, for each expression, a sequence of input tokens for the model is created by combining all the persona phrases of the current speaker with the history of the previous expressions of the dialog (usually 3 to 5 previous expressions). Furthermore, separation tokens (<BOS> and <EOS> indicating the start and the end of a sentence) are also added to further separate each expression of the dialog, a common tactic in Transformer's inputs (Radford et al., 2018; Devlin et al., 2018).

The word and positional embeddings learned during the pre-training phase, augmented with a set of dialog-state embeddings shown in Figure 20. This set of additional embeddings is utilized to indicate whether the current token is part of (i) a personality phrase, (ii) an expression from PERSON1 or (iii) an expression from PERSON2. These additional embeddings are learned during the fine-tuning process on the dialogue dataset. The input of the Transformer model's self-attention block is then the sum of the three types of embeddings (word, dialog-state and positional) for each word.

3.4.4. Fine-Tuning on Dialogue

The model uses an architecture called "Double-Head" in which the two heads are two linear layers. The language modeling head has its weights attached to the input embeddings and the classification head uses the input of the specified classification token index in the input sequence as input. Thus, one head will calculate the language modeling predictions, while the other head will predict next-sentence classification labels (Figure 21).

The fine-tuning process is accomplished by optimizing a combination of two loss functions: (i) a next-sentence classification loss, and (ii) a language modeling loss. The total loss will be the weighted sum of the language modeling loss and the next-sentence prediction loss calculated as follows:

(i) The next-sentence classification loss is used in the classifier training to distinguish the correct next utterance appended to the input sequence from a collection of randomly sampled distractor-sentences (a list of 20 sentences in which the last one is the ground truth response observed in the conversational data). The hidden-state of the last token (the end-of-sequence token) is passed through a linear layer to obtain a score and apply a cross-entropy loss to correctly classify the response among the distractors.

(ii) Project the hidden-state on the word embedding matrix to obtain logits and apply a cross-entropy loss to the part of the target corresponding to the correct response (green labels on Figure 21).

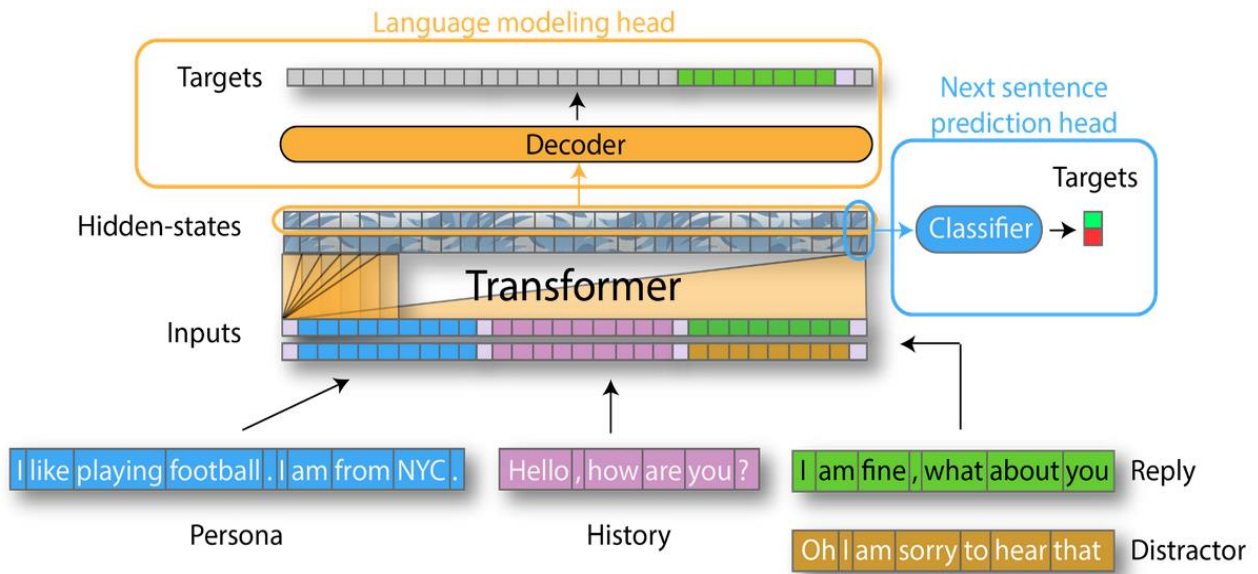


Figure 21. . Multi-task training objective - the model is provided with two heads for language modeling prediction (orange) and next-sentence classification (blue)

3.4.5. Generation Method for the Chatbot

The generation method for the conversational agent is performed using top-k and nucleus (or top-p) sampling (Ari Holtzman et al., 2020). Sampling-based decoding methods have shown a strong ability in generating diversity, fluency and repetitiveness of the generation with pre-trained language models. The general principle of these two methods is to sample from the next-token distribution after having filtered this distribution to keep only the top k tokens (top-k) or the top tokens with a cumulative probability just above a threshold (nucleus/top-p). For nucleus sampling, the key idea is to use the shape of the probability distribution to determine the set of tokens to be sampled from. Given a distribution $P(x|x_{1:i-1})$, its top-p vocabulary $V(p) \subset V$ is defined as the smallest set such that:

$$\sum_{x \in V(p)} P(x|x_{1:i-1}) \geq p \quad (7)$$

Let $p' = \sum_{x \in V(p)} P(x|x_{1:i-1})$. The original distribution is re-scaled to a new distribution, from which the next word is sampled:

$$P'(x|x_{1:i-1}) = P(x|x_{1:i-1})/p' \text{ if } x \in V(p) \quad (8)$$

otherwise:

$$P'(x|x_{1:i-1}) = 0 \quad (9)$$

In practice this means selecting the highest probability tokens whose cumulative probability mass exceeds the pre-chosen threshold p. The size of the sampling set will adjust dynamically based on the shape of the probability distribution at each time step.

3.5. REQUIREMENTS OF A MULTI-LANGUAGE CONVERSATIONAL AGENT

Language Models (LMs) based on pre-trained architectures such as BERT (*Devlin et al., 2019*) and GPT-2 (*Radford et al., 2019*) have provided impressive improvements across several NLP tasks. While for BERT-based architectures several monolingual models other than English have been developed, language-specific implementations of generative pre-trained transformer based models, such as GPT-2, are not widely available yet. However, a multi-language (English, Italian, Greek) open-domain Conversational Agent is required for the purposes of REBUILD. The implementation for English is described in section (3.4.1) where DIALOGPT is used as a pre-trained model and is fine-tuned to the PERSONA-CHAT dataset. Following the same strategy for both the Italian and the Greek language, the two main factors required for the conversational agent to be developed are: i) pre-trained models as GPT-2 in both languages and ii) a dataset as it is described in section (3.4.2) for the fine-tuning procedure in the dialogue.

Italian and Greek Pre-Trained Language Models:

Italian: The recently pre-trained model weights open-sourced by GePpeTto (*Mattei et al., 2020*) is proposed for Italian. GePpeTto uses the original GPT-2 as base and is trained on a large corpus of Italian text consisting of two main sources: Italian Wikipedia and ItWac (*Baroni et al., 2009*) corpus.

Greek: A pre-trained GPT-2 was not available for the Greek language, so we should fill this gap. Following the work of (*Radford et al., 2018*) we train from scratch a generative Transformer model as GPT-2 on a large corpus of Greek text so that the model can generate long stretches of contiguous coherent text. The model is trained on a collection of almost 5GB Greek texts, with the main source to be from Greek Wikipedia. The content is extracted using the Wikiextractor tool (*Attardi, 2012*). The dataset is constructed as 5 sentences per sample (about 3.7 millions of samples) and the end of document is marked with the string `<|endoftext|>` providing the model with paragraph information, as done for the original GPT-2 training set by *Radford*. The input sentences are pre-processed and tokenized using 22,000 merges of byte-pair encoding. Attention dropouts with a rate of 0.1 are used for regularization on all layers and L2 weight decay of 0,01. In addition, a batch size of 4 and accumulated gradients over 8 iterations are used, resulting in an effective batch size of 32. The model uses the Adam optimization scheme with a learning rate of 1e-4. The learning rate increases linearly from zero over the first 9000 updates and decreases linearly by using a linear schedule. The model is trained until there is no progress in validation loss. The implementation is based on the open-source *PyTorch-transformer* library (*HuggingFace 2019*).

All the pre-trained models (DIALOGPT, GePpeTto and the Greek one) that are used, correspond to GPT-2 small, with 12 layers (768 dimensional states and 12 attention heads) and 117M parameters (*Radford et al., 2018*).

4. CONCLUSION

This report describes the work done so far in order to develop the Digital Companion for the purposes of REBUILD. The Digital Companion consists of two types of chatbots : a rule-based and an open-domain chatbot. On the one hand, the rule-based chatbot designed to follow specific rules, will be the one that will address the potential critical issues migrants may face with a first response. If a situation can not be managed by the chatbot, the contact will be redirected to community members, volunteers or successfully integrated migrants. The issue the rule-based

chatbot had to face is the illiteracy of the user-base, which was holding a barrier to developing a single language, written communication, chatbot.

On the other hand, the open-domain chatbot will not address the critical problems of migrants, nor does it correspond to people who don't know the language. However, this open-domain conversational agent is capable of producing daily human-to-human conversations that may be useful for migrants trying to learn a new language. For the development of the multi-language (Italian, English, Greek) open-domain chatbot, the requirements are: i) pre-trained models in each language ii) datasets with dialogues for each language. Both of these requirements were only available in English, so the English chatbot is implemented as it is described in section 3.4.1. There were pre-trained language models for both English and Italian, but due to lack of availability it was needed to create one for Greek. The only requirements missing for the open-domain chatbot to be implemented in Greek and Italian are the datasets for the fine-tuning procedure in dialogue. The next step is to develop the recommendation system of the Chatbot that is described in section 3.3.

5. REFERENCES

- [1] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, Jason Weston. 2019. The Second Conversational Intelligence Challenge (ConvAI2). *arXiv preprint arXiv:1902.00098*
- [2] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, Bill Dolan. 2020. DialogPT: Large-Scale Generative Pre-training for Conversational Response Generation. *arXiv preprint arXiv:1911.00536* (<https://github.com/microsoft/DialogPT>)
- [3] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M. Smith, Y-Lan Boureau, Jason Weston. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*
- [4] Thomas Wolf, Victor Sanh, Julien Chaumond, Clement Delangue. 2019. TransferTransfo: A Transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149* (<https://github.com/huggingface/transfer-learning-conv-ai>)
- [5] Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, Quoc V. Le. 2020 Towards a Human-like Open-Domain Chatbot. *arXiv preprint arXiv:2001.09977*
- [6] Minlie Huang, Xiaoyan Zhu, Jianfeng Gao. 2020. Challenges in Building Intelligent Open-domain Dialog Systems. *arXiv preprint arXiv:1905.05709*
- [7] Jia-Chen Gu, Zhen-Hua Ling, Xiaodan Zhu, Quan Liu. 2020. Dually Interactive Matching Network for Personalized Response Selection in Retrieval-Based Chatbots. *arXiv preprint arXiv:1908.05859*
- [8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. 2017. Attention Is All You Need. *arXiv preprint arXiv:1706.03762*
- [9] Oluwatobi O. Olabiyi and Erik T. Mueller. 2019. DLGNet: A Transformer-based Model for Dialogue Response Generation. *arXiv preprint arXiv:1908.01841*
- [10] Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2018. Language models are unsupervised multi task learners. *OpenAI Blog, 1(8)*

- [11] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. 2018. Improving language understanding by generative pre-training. *Technical report, OpenAI*
- [12] HuggingFace 2019. PyTorch transformer documentation and repository.
<https://huggingface.co/transformers>, <https://github.com/huggingface/transformers>
- [13] Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Welston, Y-Lan Boureau. 2020. Can You Put it All Together: Evaluating Conversational Agents Ability to Blend Skills. *arXiv preprint arXiv:2004.08449*
- [14] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*
- [15] Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhрева, Marat Zaynutdinov. 2018. DeepPavlov: Open-Source Library for Dialogue Systems. *Conference: ACL, At Melbourne*
- [16] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, Yejin Choi. 2020. The Curious Case of Neural Text Degeneration. *arXiv preprint arXiv:1904.09751*
- [17] Ilya Kulikov, Alexander H. Miller, Kyunghyun Cho, Jason Weston. 2018. Importance of a Search Strategy in Neural Dialogue Modelling by Ilya. *arXiv preprint arXiv:1811.00907*
- [18] Kenton Murray, David Chiang. 2018. Correcting Length Bias in Neural Machine Translation. *arXiv preprint arXiv:1808.10006*
- [19] Yilin Yang, Liang Huang, Mingbo Ma. 2018. Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation. *arXiv preprint arXiv:1808.09582*
- [20] Angela Fan, Mike Lewis, Yann Dauphin. 2018. Hierarchical Neural Story Generation. *arXiv preprint arXiv:1805.04833*
- [21] Jason Weston, Emily Dinan, Alexander H. Miller. 2018. Retrieve and Refine: Improved Sequence Generation Models For Dialogue. *arXiv preprint arXiv:1808.04776*
- [22] Vinyals, O., and Le, Q. 2015. A Neural Conversational Model. *arXiv preprint arXiv:1506.05869*
- [23] Serban I. V., Sordoni A., Lowe R., Charlin L., Pineau J., Courville A., Bengio Y. 2016. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. *arXiv preprint arXiv:1605.06069*
- [24] Serban I. V., Sankar C., Germain M., Zhang S., Lin Z., Subramanian S., Kim T., Pieper M., Chandar S., Ke N. R., Mudumba S., de Brebisson A., Sotelo J. M. R., Suhubdy D., Michalski V., Nguyen A., Pineau J., Bengio Y. 2017. A Deep Reinforcement Learning Chat-bot. *arXiv preprint arXiv:1709.02349*
- [25] Yishu Miao, Lei Yu, Phil Blunsom. 2015. Neural Variational Inference for Text Processing. *arXiv preprint arXiv:1511.06038*
- [26] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation, 43(3):209–226*.
- [27] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jacob G. Simonsen, Jian-Yun Nie. 2015. A Hierarchical Recurrent Encoder-Decoder For Generative Context-Aware Query Suggestion. *arXiv preprint arXiv:1507.0222*

- [28] Jiwei Li, Dan Jurafsky. 2016. Neural Net Models for Open-Domain Discourse Coherence. *arXiv preprint arXiv:1606.01545*
- [29] Jiwei Li, Will Monroe, Tianlin Shi, Sebastien Jean, Alan Ritter, Dan Jurafsky. 2017. Adversarial Learning for Neural Dialogue Generation. *arXiv preprint arXiv:1701.06547*
- [30] Jiwei Li, Will Monroe, Dan Jurafsky. 2016. A Simple, Fast Diverse Decoding Algorithm for Neural Generation. *arXiv preprint arXiv:1611.08562*
- [31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. 2019. BERT: Pre-training of deep bi-directional transformers for language understanding. *NAACL 2019*.
- [32] Dan Hendrycks, Kevin Gimpel. 2018. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*
- [33] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. 2003. A neural probabilistic language model. *Journal of machine learning research, 3(Feb):1137–1155*
- [34] He Bai, Peng Shi, Jimmy Lin, Luchen Tan, Kun Xiong, Wen Gao, Jie Liu, Ming LI. 2020. Semantics of the Unwritten. *arXiv preprint arXiv:2004.02251*
- [35] Vighnesh Leonardo Shiv, Chris Quirk, Anshuman Suri, Xiang Gao, Khuram Shahid, Nithya Govindarajan, Yizhe Zhang, Jianfeng Gao, Michel Galley, Chris Brockett, et al. 2019. Microsoft icecaps: An open-source toolkit for conversation modeling. *ACL*.
- [36] A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. 2017. ParlAI: A dialog research software platform. *In Proceedings of the 2017 EMNLP System Demonstration*.
- [37] Giuseppe Attardi. Wikiextractor. 2012. <http://attardi.github.io/wikiextractor>.
- [38] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, Jason Weston. 2019. Real-time Inference in Multi-sentence Tasks with Deep Pretrained Transformers. *arXiv preprint arXiv:1905.01969v1*
- [39] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. 2015. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. *arXiv preprint arXiv: 1506.06724*.
- [40] Lorenzo De Mattei, Michelle Cafagna, Felice Dell'Orletta, Malvina Nissim, Marco Guerini. 2020. GePpeTto Carves Italian into a Language Model. *arXiv preprint arXiv:2004.14253*
- [41] Ilya Sutskever, Oriol Vinyals, Quoc V. Le. Sequence to Sequence Learning with Neural Networks 2014. *arXiv preprint arXiv: 1409.3215*
- [42] Hannah Rashkin, Eric Michael Smith, Margaret Li, Y-Lan Boureau. 2019. Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset. *arXiv preprint arXiv:1811.00207*
- [43] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv: 1409.0473*
- [44] David R. So, Chen Liang, Quoc V. Le. The Evolved Transformer 2019. *arXiv preprint arXiv: 1901.11117*
- [45] Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., and Robinson, T. 2013. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *arXiv preprint arXiv:1312.3005*
- [46] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*
- [47] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, Jason Weston. 2019. Wizard of Wikipedia: Knowledge-Powered Conversational agents. *arXiv preprint arXiv:1811.01241*



REBUILD

ICT-enabled integration facilitator and life rebuilding guidance

Deliverable: D4.4 Designing the digital companion

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 822215.