https://doi.org/10.31449/inf.v42i3.2173 Informatica **42** (2018) 515–525 **515**

Automated Self-Learning Chatbot Initially Built as a FAQs Database Information Retrieval System: Multi-level and Intelligent Universal Virtual Front-Office Implementing Neural Network

Alessandro Massaro, Vincenzo Maritati and Angelo Galiano
Dyrecta Lab, Via V. Simplicio 45, 70014 Conversano (BA), Italy
http://www.dyrecta.com
E-mail: alessandro.massaro@dyrecta.com, vincenzo.maritati@dyrecta.com, angelo.galiano@dyrecta.com

Keywords: chatbot, artificial intelligence, neural networks, multi-level self-learning information system.

Received: February 2, 2018

The method proposed in this paper is based on dynamical information system capable toimplement a universal multi-level virtual front-office made by FAQs and chatbot self-learning systems.We describe statistics and tests necessary to validate the solution, and report acomparison between neural network and AIML results.

Povzetek: V prispevku je opisana nova metoda za izdelavo virtualnega asistenta iz arhiva vprašanj in odgovorov.

1 Introduction

Robot and artificial intelligence appeared a lot of time ago in the design of the "*Leonardo's mechanical knight*" [1] and earlier in the 12th century by Al-Jazari [2]. Artificial Intelligence (AI) begins eight centuries later with the conceptualizations of Alan Turing (Turing's test) [3]. The merging of robot and artificial intelligence is much more recent and opens perspectives with very strong potentials and related concerns. Virtual assistants as "*Chatbot*" (also known as a talkbot, chatterbot, Bot, IM bot, interactive agent, or Artificial Conversational Entity) can dialogue as real personal assistants; their main current use is on smartphones [4] to retrieve information useful for everyday life, to set the alarm or an appointment on the agenda, to send mail or text messages, find places or browse or launch apps. As robots enables mechanical automatisms, chatbot implementing AI allows the information automatism.

A chatbot is a computer program designed to simulate conversation with human users, especially over Internet; it acts like a human computer interface created to facilitate communication between human and computer, understanding natural language questions and answering with actual answers. Although chatbot is a current hot topic, it has been object for the past fifty years. The chatbot systems idea originated in M.I.T back in 1966, where professor Joseph Weizenbaum implemented the ELIZA chatbot to emulate a psychotherapist [5]. After ELIZA have been developed a lot of chatbots, for example to simulate the interaction with different personalities [6], matching with web-based search engines (AskJevees) [7], and with open-source initiatives like ALICE [8] [9] implementing artificial intelligent applications called AIML (Artificial Intelligence Markup Language) [8]. AIML is a widely adopted standard for creating chatbots and mobile virtual assistants like ALICE [8], Mitsuku [10], English Tutor [11], The Professor [12] and many more. Over the years, chatbots have become a sophisticated tool, able to perform natural conversations and make users happy for the quick support they can provide. Although a chatbot cannot handle all customer queries, it can be used to deal with many of the routine queries activating service requests.

The knowledge bases of existing chatbots are mostly built manually [13], thus requiring a long time, and are difficult to adapt to new domains. There are several researches on the extraction of knowledge from different types of data sets [14] [15] [16] but these approaches use the characteristics of their domains and are therefore only suitable for their specific tasks, limiting the possibility to transform directly their methods in a general knowledge extraction approach.

Nowadays there are tens of thousands chatbots available online, over 30K already on the Messenger platform alone [17], and their number is growing rapidly thanks to the ease of implementation and distribution provided by services such as Facebook Messenger, Telegram, etc. Users around the world are logging into messaging apps to not only chat with friends but also to connect with brands, to browse merchandise, and to watch content (in the mid-2014 the number of messages exchanged on the main four platforms of instant messaging has exceeded the number of messages exchanged on the 4 main social networks [18]).

The approach described in this paper involves the implementation of a smart virtual front-office model that exploits the synergy between a list of FAQs and a chatbot [19], [20]. The main features of this model are:

- be able to self-learn their knowledge base from an archive of questions and answers (FAQs) organized in a static tree structure;
- the use of machine learning algorithms to dynamically generate its own knowledge base;
- the speeding up of the management of most user requests;
- the possibility to acquire different feedbacks thus improving the efficiency of the system, so as to be able to easily reshape the available data according to greater effectiveness in supporting users;

- allowing full control over the management activities of the system itself.

Microsoft today offers a service loading knowledge base (KB) for a chatbot by copying and pasting phrases of FAQs [21]. Recently some researchers implemented long short-term memory (LSTM) neural networks [22], which automatically generate responses for users requests on social media. Other authors highlighted that a chatbot does not understand colloquial usage [23], and cannot yet simulate the full range of intelligent human conversation

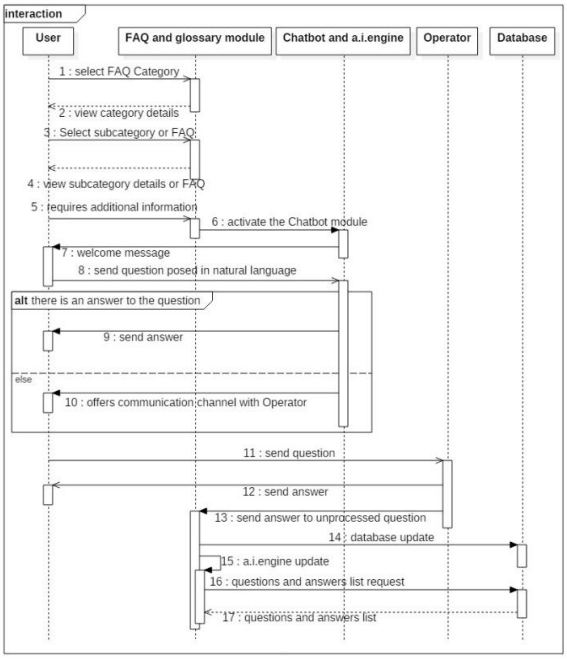

Figure 2 - System's interactions

[24].

In the proposed research are presented the following claims with respect to the state of the art:

-automatic loading and construction of the knowledge base of the chatbot system (automatic generation of the AIML files constructing the KB by an automatic data entry);

-automatic updating of the training dataset, progressively reducing the probabilistic error of request recognition (see self-learning multi-level model enabling automatic storing of the training dataset described in the next section);

-gradually overcoming the barrier due to the training of model including colloquial usage forms.

2 Case study

The proposed case study consists of improving customer support through a website by exploiting the FAQs archive available to its users, and useful for the automated creation of a chatbot. The management of requests for assistance from a chatbot will reduce the workload of human operators dedicated to customer support and will allow the assistance service to be operational even in the absence of available operators.

If the system does not find a response to a user question, it can redirect the user to a human operator or store the user questions and data in order to send these information to the first available operator.

Figure 1 shows a functional scheme of the whole system.

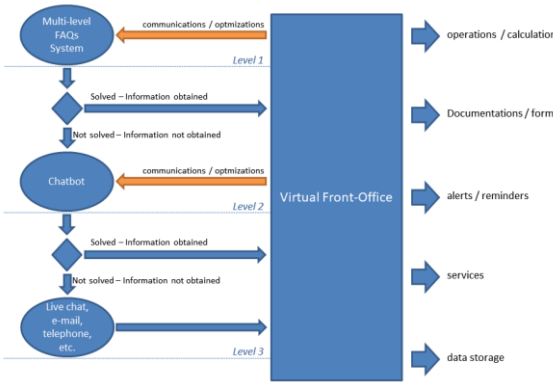

Figure 1 - Functional scheme

The multi-level access to the different modules of the system allows the optimization of available resources and, at the same time, a better user-experience:

- according to the sections consulted by users, a series of relevant FAQs, organized in a tree structure, are proposed (***level 1***);
- when the user requests a contact with an operator to ask further information, he is put in touch with the chatbot that, according to the user's requests and related information, proposes the most relevant answers (***level 2***);
- if the chatbot does not find a response for the user, it puts him in contact with a specific human operator, selected by the system on an available operators list, on one of the available channels (e.g. e-mail, chat, telephone – ***level 3***), thus optimizing the response time; the response provided by the operator is memorized by the system and is exploited by a self-learning algorithm to increase the knowledge base of the system.

Operators are able to improve the knowledge base and the overall performance of the system taking advantage of:

- the archive of questions to which the chatbot does not respond (the training dataset is gradually enriched by adding a new FAQ formulated through of the three level responses if Fig. 1);
- a series of tools, tests and indications provided by the system in a fully automatic way in order to make the operators capable of optimizing the knowledge base (operators is guided by a platform allowing an automatic update of the training dataset).

The system’s interactions are sketched in the diagram of Figure 2.

The diagram of Figure 3 represents the interactions between the chatbot module and the artificial intelligence engine.

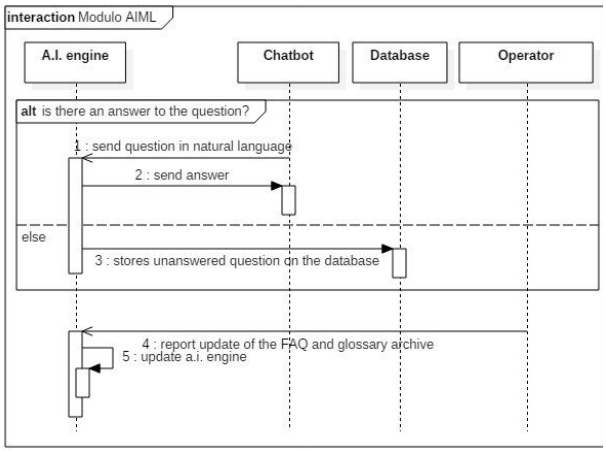

Figure 3 - The interactions between the Chatbot and the artificial intelligence engine

3 System design

The system has been designed in a modular way to allow fast integration into any web-based platform and isn't bound to a particular environment. It can be logically divided into three macro-modules:

- FAQ: management by authenticated users (operators and administrators) and display by unauthenticated users of FAQs and Glossary items;
- Chatbot: management and use of the "Chatbot" application that will allow interaction between users and the system by simulating communications in natural language;
- Back-office: management of data and the consequent information in the system available to users; system supervision; statistics; system optimization.

These three macro-modules are detailed in the use case diagram of Figure 4, where there are explained the main actions performed; this diagram shows the actors of the system:

- the end user, which accesses to the FAQ and chatbot system to find answers to some of his questions;
- the operator of the system that has access to the back-office interfaces for managing the information that the system makes available to the users;
- the system administrator who has access to back-office interfaces to supervise the system's activities.

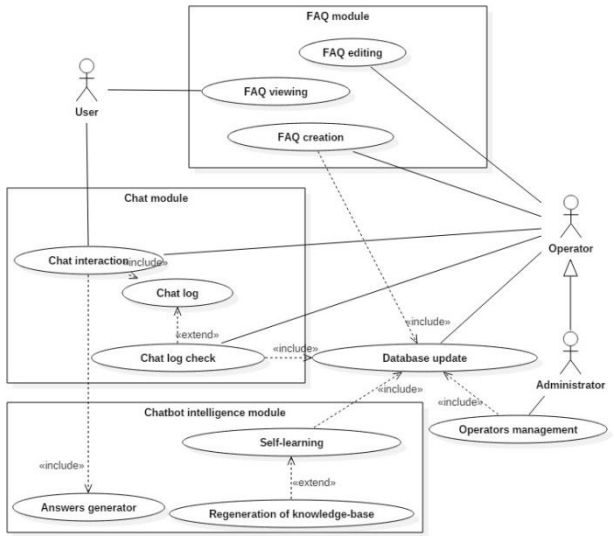

Figure 4 - Use case diagram

The database is used as aggregator of the data coming from the system modules and provides the necessary information for data processing. In this way it is possible to decouple the logic operations from the data administration, as a function of a smart management of all the underlying functionalities.

During a user's visit, the system stores its preferences and the sections visited so that these information can be used to better evaluate which answers to generate to the user's questions.

Each component of the system cooperates synergistically with the others in order to improve the user-experience; at the same time this integration allows to update the system's knowledge base by simply operators tools, exploiting an archive of questions to which the system has not been answered, and providing useful indications on how to improve the knowledge base in function of a better performance.

The diagram of Figure 5 shows the interaction between the system components.

The **autoresponder with self-learning module** consists of two modules: an artificial intelligence (A.I.) module, and a chatbot module; the A.I. module generates the answers of the user questions, and allows to train the knowledge model. Both modules are connected to the database but only the chatbot module modifies it, by inserting new data useful for statistical purposes and for the optimization of the knowledge base. The use of machine learning algorithms based on neural networks has been compared with the use of AIML during the design of the artificial intelligence part of this module, in order to validate the proposed final solution. The chatbot module also manages user questions and system-generated responses to make them available to administrators for control purposes, and to generate statistics on system usage. An algorithm has been modeled to read the knowledge base available in the database and to transform these data into a format that is exploitable by neural networks able to generate AIML patterns.

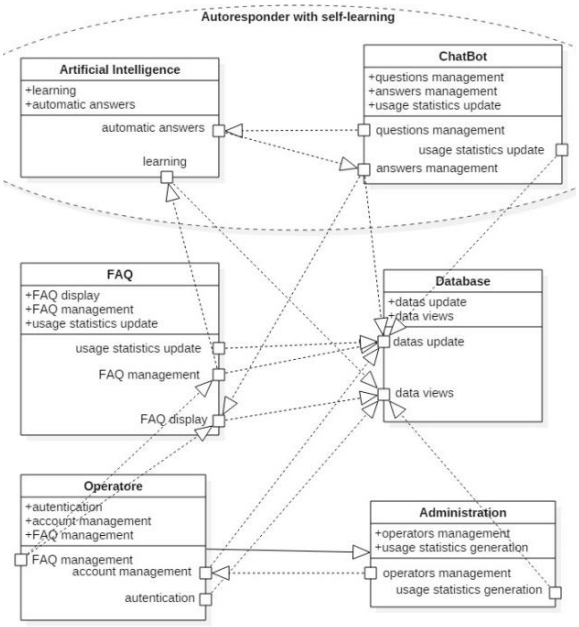

Figure 5 - Composite structure diagram

The database module deals with exchanging data with the database server by recycling and caching queries in order to optimize the performance; this module provides a single interface for all modules of the system connected to the database.

The FAQ module allows the complete management of the FAQs and glossary entries; this module updates the system usage statistics regarding the operators activities.

The operator module manages all the actions concerning the operators of the system, by allowing the authentication of each operator or administrator, the modification of his personal data and, in the case of users authenticated as administrators, the management of other accounts through the administrator module; the operator and the administration write on the database all the actions performed by a user for statistical purposes.

3.1 Data model

The entity-relationships model of Figure 6 highlights the data necessary for each system entities and how these are associated with each other.

This simplified diagram shows a clearer description of the main characteristics of the database architecture.

From Figure can be deduced the following data structure:

- FAQs are grouped into categories that can be grouped into other parent categories themselves, creating a tree structure that simplifies the selection and use of FAQ data;
- every user can have different roles in the system (*tables: User, UserRole* and *Role*)
- there may be more different answers to a question and

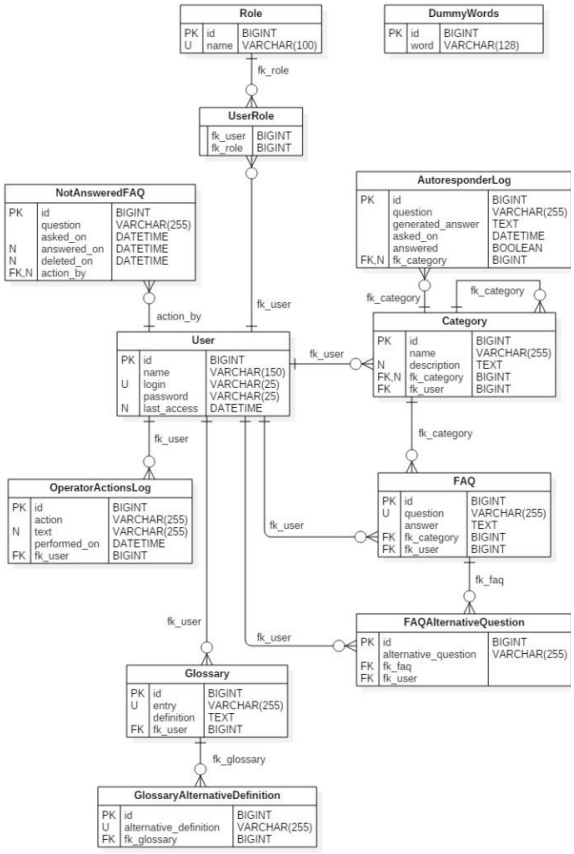

Figure 6 - Entity-relationship diagram

there may be different definitions for a glossary entry (*tables: FAQAlternativeQuestion* and *GlossaryAlternativeDefinition*);

- words without semantic content are stored in a specific table (*DummyWords*);
- all the users' questions to which the chatbot has not answered are traced (*table: NotAnsweredFAQ*); the operators can use these questions to create new FAQs or to add new answers to existent FAQs;
- all the chatbot answers and the corresponding user questions are traced (*table: AutoresponderLog*);
- all the actions performed by the operators on the system are traced (*table: OperatorActionsLog*) for statistical and control purpose.

The dynamic training dataset of the chatbot system (initial knowledge base) is built from the previous described data model: the dataset is constructed by a starting database containing glossary, by the FAQs database, and by the *DummyWords* list (words to ignore).

3.2 Self-learning algorithm

Upon receipt of a natural language question asked by a user, the first operation to be performed is the elimination of all words without semantic meaning according with the context in which the chatbot operates; this operation is performed through a comparison with a list of words, characters and symbols to be ignored.

Once the words without semantic meaning are removed from the original question, a list of remaining keywords remains is passed to the A.I. engine thus activating the answer searching process.

If the A.I. engine finds a response to the keyword sequence, this is sent to the user; otherwise the question is stored to be subsequently verified by a human operator. According to the last user choices and the context in which the chat has been opened, the question is passed to a specific operator.

When an operator answers to an user question, the response provided by the human operator is stored, and it is used to regenerate the knowledge-base.

This algorithm depicted in the next diagram of Figure 7.

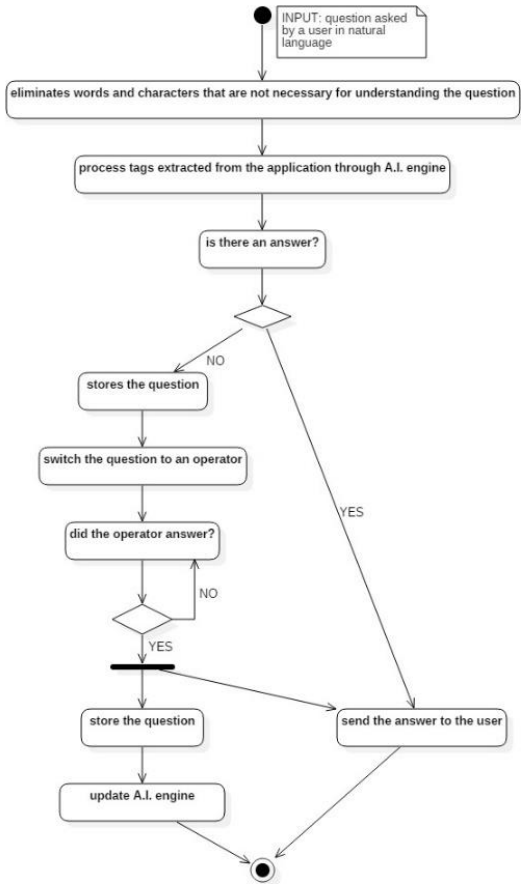

Figure 8 - Self-learning algorithm

Second action of the previous flow-chart "process tags extracted from the application through A.I. engine" underlies two A.I. pre-processing algorithms: one for the automatic production of standard AIML patterns, and one for setting the right weights of the neural network. Pre-processing algorithms take into account user question generating outputs for feeding to the respective A.I. engine. The reply of these engines has the same data type (a natural language phrase and a Boolean value indicating response validity).

The AIML A.I. engine is based on an open-source library compatible with AIML 1.0.1 standard. The artificial neural network (ANN) is a fully connected neural network that is used as a universal functions approximator performing the training, and behaving as a classifier of questions.

3.2.1 Automated AIML patterns production

The automatic production of AIML patterns based on system knowledge-base (FAQs, glossary entries, dummy words), is managed by an iterative algorithm that extracts keywords from the input text and using them to construct AIML patterns (see Figure 8). All the generated AIML patterns are stored in a file ready to be processed by the AIML engine.

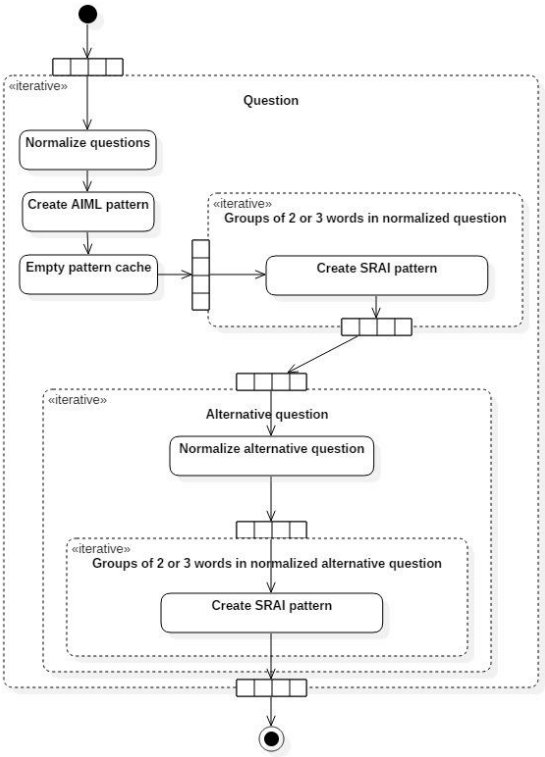

Figure 7 – Automatic AIML patterns creation

For the automatic reconstruction of AIML files, all the questions (between FAQ, Glossary and alternative questions of both) are analyzed in a cycle that, for each question:

1. normalize the question in the archive (remove words with no semantic meaning and unwanted characters such as punctuation, dashes, etc.);
2. create an AIML pattern on the normalized question;
3. clean the AIML pattern cache;
4. for each group of 2 and 3 words taken from words in the normalized question of point (1):
   - create a Symbolic Reduction in Artificial Intelligence (SRAI) pattern on the AIML pattern;
   - verify that the SRAI pattern has not already been created based on the AIML pattern cache;
5. for each alternative question to the current question:
   - normalize the alternative question, as for the original question;

- for each group of 2 and 3 words taken from words in the normalized alternative question:
  - create a SRAI pattern on the AIML pattern of the original question;
  - verify that the newly created SRAI pattern has not already been created based on the AIML pattern cache.

This process is started automatically when an operator changes the knowledge base of the system.

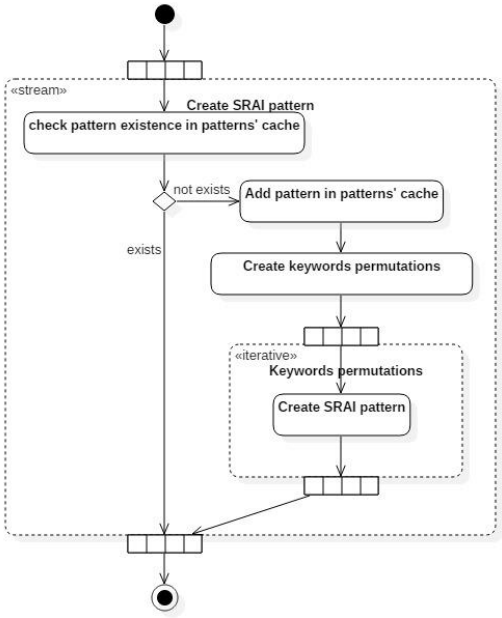

Figure 9 - SRAI pattern creation

It is possible to disable the automatic creation of AIML patterns in order to allow massive changes to the knowledge base and to re-enable this functionality at the end of all changes: in this way the operators avoid making unnecessary reconstruction of the AIML patterns until all the necessary changes have been made to the knowledge base.

The diagram in Figure 9 exhibits the steps for the creation of each SRAI pattern related to each original question and to all the alternative questions generated by the original question.

By compared Figure 9 with Figure 8, the only passage to be explained in this diagram is the "Create keywords permutations" action, having the task to create the SRAI pattern in different combinations by the 2 or 3 selected keywords using the wildcard character [*] provided by the AIML standard.

3.2.2 AIML optimizer

In order to find the best combination of patterns for the construction of the AIML file that will manage the chatbot, an automatic algorithm has been implemented. The chatbot:

- is able to find the best combination of keywords / patterns among all those that can be created;
- can find any redundant questions, which lead to different answers while being syntactically similar (these questions are reported to the operators so that they can verify and modify them according with an appropriate knowledge base).

This algorithm is called for each editing process in the knowledge-base. The AIML optimizer algorithm is described in the next flow-chart diagram of Figure 10, where it is highlighted the optimization procedure that aims to remove some keywords between those obtained in the analyzed question thus activating the system response.

3.2.3 AIML vs neural network architecture

An AIML module is based on the recognition of precise textual patterns. By using a defined logic in the patterns definition, the AIML allows the construction of recursive patterns and the use of wildcards that subtend one or more words, making AIML a standard ready to simulate a conversation in natural language.

Although the process of creating AIML patterns can be automated, the final result is a series of static patterns, which will respond to the inputs according to precise rules and structures.

Better results can be achieved using an artificial neural network instead of an AIML-based system, because:

- there will be a better correspondence between the questions proposed to the system and the answers provided by it (depending on the implemented artificial neural network architecture, it is possible to construct complex maps between the input and the output text generated by the system, exceeding the limits and the static nature of AIML patterns and handling specific or difficult cases);
- the execution speed is greater, especially in conjunction with large archives of texts to be managed (while the number of texts to be managed increases, the number of AIML patterns necessary to manage all the possible cases increases; an artificial neural network does not need the addition of new neurons or new layers in order to manage a new texts; moreover the recognition of AIML patterns occurs through the direct comparison of text strings which is a very slow operation, especially when compared to the additions and multiplications used in neural networks, operations in which the electronic computers excel thus requiring parallel computing). In Figure 10 is illustrated the flowchart of the AIML optimizer.

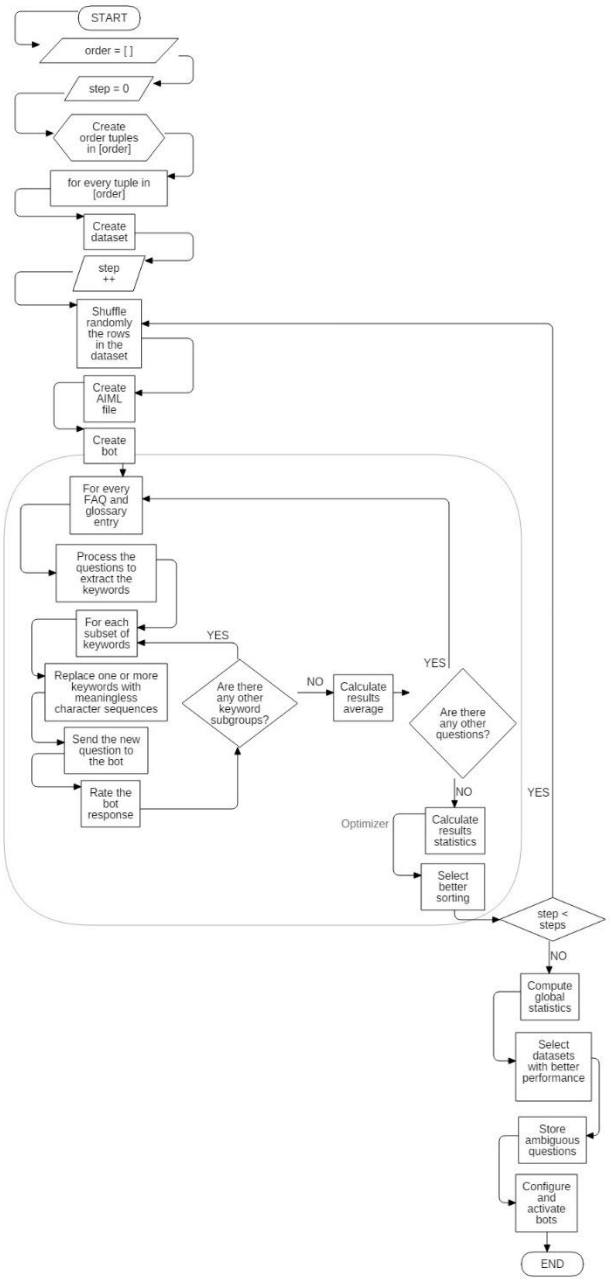

Figure 10 - AIML optimizer

3.2.4 Automated neural network

The automatic creation of the neural network has been implemented by exploiting some algorithms able to process the knowledge base of the system without the intervention of human actors. The main parameters of the neural network, like the number of neurons used in the hidden layers, are automatically set on the basis of the available dataset, so as to allow an adaptation of the solution to any new dataset suitable for the system implementation.

The technique called "one hot encoding" [25] (discussed in the following paragraph) has been adopted to make the text of the questions usable by the underlying neural network. Mathematically, one hot encoding generates a balanced matrix, which is easy to understand during complex computations inside algorithms. One hot encoding technique is then used to encode categorical features: one-hot is a group of bits among which the allowed combinations of values are only those with a single high bit (1), and all the others are low (0) [25].

3.2.4.1 One-hot encoding

The words found in the questions archive, allow to create a dictionary. In this dictionary, each word will correspond to an index. In order to transform any new sentence into binary vectors, a binary vector is created having a length equal to the number of words of the dictionary and having all the elements equal to zero. Below is shown an example useful to understand the one-shot encoding approach.

For example, taking the following three sentences:

```
[ today is a beautiful day ]
[ she is a beautiful girl ]
[ that girl has a red car ]
```

is created the following dictionary of 11 words:

```
[today,is,a,beautiful,day,she,girl,that
,has,car,red]
```

A new sentence like:

```
[ your new red car is simply beautiful ]
```

would be converted to the following binary vectors:

```
[ 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1 ]
```

which turned into text would be for the four weighted keywords:

```
[ new beautiful car red ]
```

The semantic meaning of the sentence has been lost during the transformation, but a fully connected ANN processing this input can still find a combination of its weight values to match the best answer to each question ignoring the order in which the words of the sentence are placed: the combination of words is processed by taking into account only the maximum weight of their occurrence into a sentence.

Another way is to create a binary vector for each word of the input question and thus create a binary matrix able to bring back the order in which the single words succeed each other in the original question.

The `[ your new red car is simply beautiful ]` sentence would be converted to the following binary array:

```
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 ]
[ 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 ]
```

where the units represent the keyword presence structured in a predetermined sized matrix, allowing also to extract information of the word position into the sentence (important aspect for colloquial usage forms). In this matrix each line corresponds to a word of the sentence, and the lines are ordered following the order of the analyzed sentence. In each row, the position of a "1" indicates which word to consider in the sentence sequence

This array make possible to quickly process the relative textual information. If the array is processed by the recurrent neural network, will be also possible to analyze the order of the input keywords.

To obviate the problem of the management of sentences composed by different number of words, it is used the technique of zero-padding: by this approach are added null vectors having a number equal to the number of missing words up to a value N, where N is the number of words managed; if a sentence has a number of words that exceeds the limit of the words managed, only the first N words will be considered for the data processing.

3.2.4.2 Neural network based chatbot

In our model a normalization function on the texts to be processed is considered. This function clean the words with no semantic meaning and not useful for the project aims. All input texts are cleaned by articles, adverbs, prepositions, special characters, accented letters, etc. in order to analyze the input text exclusively by the key words useful in tracing the most appropriate answer for the input question.

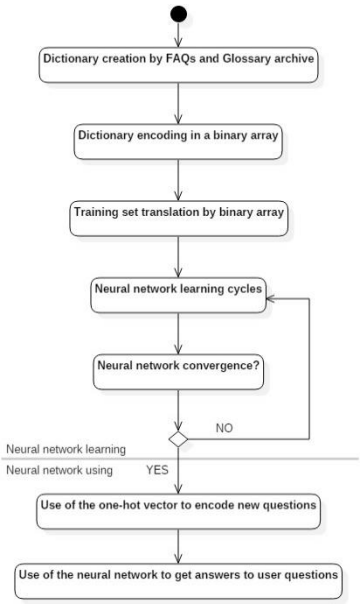

Figure 11 - Neural network learning and using model

In Figure 11 is illustrated the diagram of the used model.

The learning phase (upper part of Figure 11) is started at each database variation made by an operator updating the questions/answers knowledge base. The learning phase duration depends on the number of questions to be analyzed, but is faster in the execution if compared with the equivalent algorithm of the AIML patterns creation.

Another advantage in the execution time is also checked when the neural network is used to answer new questions.

3.2.4.3 Neural network model

The ANN model used in the project is reported in the following Python script based on the Keras library:

```
  model = Sequential()
  model.add(Dense(input_features,
activation='relu',
input_dim=input_features ))

model.add(Dense(int(input_features/1.5),
activation='relu'))

model.add(Dense(int(input_features/1.5),
activation='relu'))
  model.add(Dense(int(input_features/2),
activation='relu'))
  model.add(Dense(int(input_features/2),
activation='relu'))
  model.add(Dense(int(input_features/3),
activation='relu'))
  model.add(Dense(int(input_features/3),
activation='relu'))
  model.add(Dense(output_class,
activation='softmax'))
  model.compile(loss='categorical
crossentropy', optimizer='nadam',
metrics=['accuracy'])
```

The main characteristics of the proposed model are:

- parameterization of the number of neurons for each layer based on the dictionary size;
- use of the ReLu activation function to speed up the learning process [26], [27];
- use of Nesterov Adam optimizer [28] to obtain a faster convergence of the neural network;
- reduction of the number of neurons per layer to induce the ANN to behave like an encoder able to classify the questions according to the available answers.

4 Tools and development

The system is based on Python 3.x, using the Django web framework. The database is built in MySQL Community Edition 5.7.

The Python libraries used are:

- Keras - deep learning library;
- Tensorflow - an open source software library for Machine Intelligence;
- MySQLdb - thread-compatible interface to MySQL database server;
- NumPy - a package for scientific computing;
- PyAIML - an interpreter package for AIML.

Some features of the system are performed by exploiting Jupiter Notebook on an Anaconda distribution; this development method allows to quickly test thus validating some modules of the system before inserting them as components of the final prototype system (preliminary test phase).

Eclipse Oxygen was used as IDE, with PyDev plugin installed to support Django's modules development.

Some useful functions are adopted to test the system and to optimize the underlying dataset. By using these functions, it is possible to evaluate the efficiency of the AIML Chatbot system vs Neural Network Chatbot system without an huge dataset of new questions to be submitted to the system.

The most relevant function for the optimization of the dataset used for the learning stage of the A.I. module, is a function that adds random noise in the questions used for training and that uses the new produced questions to verify the effectiveness of A.I. algorithm. This function is based on the optimizer of the algorithm described in 3.2.2 where the only difference is that the ANN is used instead of the AIML engine for the evaluation of the chatbot performance.

Comparing the answers generated by the system on all the questions in the archive, and modifying these questions in different ways, it is possible to construct a metric able to indicate the redundancy degree of the questions sored into the archive. Staring to the results provided by this function, it is possible to create visual interfaces for operators able to highlight any questions to be modified. These interfaces are useful to build a more appropriate knowledge base.

The following table shows the average results obtained by applying the described algorithm applied on 10 datasets of questions and answers concerning 5 different application domains; each dataset includes 180÷300 questions and answers whereas a set of questions can be referred to a single answer:

Keeped Keywords	Accuracy	Mistaked Answers	No reply
All	98,925%	0%	1,075%
None	0%	0%	100%
50%	37,634%	3,226%	59,14%
66,6%	83,871%	0%	16,129%
75%	89,247%	1,075%	9,677%
80%	98,925%	0%	1,075%
83,3%	98,925%	0%	1,075%
85,71%	98,925%	0%	1,075%

where “all keeped keywords” refers to the results of the original dataset, the “none keeped keywords” refers to the obtained results skipping every keywords (the system has not knowledge base), the “50% keeped keywords” refers to the obtained results skipping a keyword every two keywords, the “66,6% keeped keywords” refers to the obtained results skipping a keyword every three keywords and so on.

Analyzing the questions to which the system responded by proposing an incorrect answer, it is possible to make a correction on the same questions, exploiting the keywords highlighted as equivocal to replace these questions with similar ones but based on different keywords. The analysis of the unanswered questions generated by the system provide an indication to the operators on which topics to generate new questions and / or answers in order to enrich the system knowledge base in an optimized way. The accuracy index provides a useful value to compare the system's performance before and after any substantial changes to the knowledge base so as to report to the operators any final performance decays.

5 Results

The approach described in this document allows to obtain a highly automated management system of front-office services applicable in any context characterized by a knowledge base consisting only of an archive of questions and answers like a FAQs archive. The main advantage in adopting this approach is given by the high automation of the different processes underlying the system, which allows:

- an easier management of front-office procedures by implementing an efficient multi-level architecture able to automatize the self-learning training process;
- an easier data entry procedure interfaced with a portal framework (the user adds requests online into a web page thus enriching automatically the training dataset and eliminating other manual steps for the dataset construction);
- a lower workload for operators, enabled by the chatbot that replaces operators in most users' questions (when the training dataset will be efficient the operators will be totally theoretically replaced by the virtual assistant);
- the applicability in any context, not dependent on the technologies used (the self-learning process could be applied in different applications where users require responses);
- the usage of the chatbot system also for colloquial usage forms.

The comparison between AIML and ANN for the automatic construction of chatbot based on the same input datasets, allowed to verify the best performances of the second approach. Using the metrics described in the previous paragraph it has been possible to find a better accuracy, a smaller number of wrong answers and a smaller number of missed answers through the use of an ANN through to the use of AIML, as shown in the following table:

Artificial intelligence module	Accuracy	Mistaked Answers	No reply
AIML	85%	5%	10%
Fully connected neural network	99%	0%	1%

An improvement to the proposed solution may be provided by the use of a recurrent ANN, in order to exploit the memory of the neural network to better associate the responses available in the system to the questions of its users, thus recognizing the order in which the keywords are arranged in the sequences created from a user’s question.

6 Conclusion

Authors proposed in this work an innovative self-learning multi-level virtual front office, by improving a structured FAQ procedure together with an innovative chatbot system. The proposed model is suitable for industrial applications requiring the optimization of human resources activities. The goal of the self-learning model is

to eliminate totally the humans responses represented by level 3 of the model of Fig. 1, by constructing dynamically and automatically the training dataset. The model could be applied to Big Data and Machine to Machine (M2M) systems [29]. The experimental dataset is reported in [30].

7 Acknowledgement

The work has been developed in the frameworks of the Italian projects: "*Piattaforma universale multilivello di front office virtuale intelligente*" (Universal multi-level platform of intelligent virtual front office) -Multi-Level Virtual Front Office-. Authors gratefully thanks V. Antonacci, L. D'Alessandro, G. Lonigro, G. Ronchi, G. Sicolo and E. Valenzano. Special thanks are addressed to N. Calamita and P. Re David for their support in the definition of the application's scenario.

8 References

[1] M. E. Rosheim (2006) "Leonardo's Lost Robots" Springer-Verlag Berlin Heidelberg 2006.

[2] W. Aguilar, G. Sanatamaria-Bonfil, T. Froese and Carlos (2014) "The Past, Present, and Future of Artificial Life" Frontiers in Robotics and AI, Vol. 1.

[3] Turing, A. (1950). "Computing Machinery and Intelligence" Mind vol. LIX N° 236. https://doi.org/10.1093/mind/LIX.236.433

[4] S. V. Doshi, S. B. Pawar, A. G. Shelar and S. S. Kulkarni (2017) "Artificial Intelligence Chatbot in Android System using Open Source Program-O" International Journal of Advanced Research in Computer and Communication Engineering, Vol. 6 (4), 816-821. https://doi.org/10.17148/IJARCCE.2017.64151

[5] Weizenbaum, J. (1966) "ELIZA - A Computer Program for the Study of Natural Language Communication between Man and Machine" Communications of the Association for Computing Machinery 9: 36-45. https://doi.org/10.1145/365153.365168

[6] Colby K.M. (1999) Human-Computer Conversation in A Cognitive Therapy Program. In: Wilks Y. (eds) Machine Conversations. The Springer International Series in Engineering and Computer Science, vol 511. Springer, Boston, MA. https://doi.org/10.1007/978-1-4757-5687-6_3 AskJevees. (2004). [Online]: https://uk.ask.com/

[7] Wallace, R. (2004). "The elements of AIML style." ALICE AI Foundation.

[8] R. S. Wallace (2003). [Online]: http://alice.pandorabots.com by ALICE AI Foundation

[9] Steve Worswick (2003). [Online]: http://www.mitsuku.com by Square Bear.

[10] English tutor (2011). [Online] https://www.eslfast.com/robot/english_tutor.htm by ESL Robot.

[11] The Professor (2012). [Online]: https://www.pandorabots.com/pandora/talk?botid=935a0a567e34523c by ALICE AI Foundation.

[12] J. Huang, M. Zhou and D. Yang (2007) "Extracting Chatbot Knowledge from Online Discussion Forums" IJCAI'07 Proceedings of the 20th international joint conference on Artifical intelligence, 423-428.

[13] L. Shrestha and K. McKeown (2004). "Detection of question-answer pairs in email conversations." In Proceedings of Coling, pp.889-895. https://doi.org/10.3115/1220355.1220483

[14] R. Nishimura, Y. Watanabe and Y.Okada (2005). "A Question Answer System Based on Confirmed Knowledge Developed by Using Mails Posted to a Mailing List." In Proceedings of the IJCNLP 2005, pp.31-36.

[15] L. Zhou and E. Hovy (2005). "Digesting Virtual Geek Culture: The Summarization of Technical Internet Relay Chats" In Proceedings of ACL 2005, pp.298-305. https://doi.org/10.3115/1219840.1219877

[16] Edelman Digital's (2016). "2017 Trend Report" [online] https://edelmandigital.com/wp-content/uploads/2016/12/2017-Edelman-Digital-Trends-Report.pdf

[17] Business Insider (2015). "Messaging apps are now bigger than social networks" [online] http://uk.businessinsider.com/the-messaging-app-report-2015-11

[18] A. Shawar, B.A. Atwell, A. Roberts (2005). "FAQchat as in Information Retrieval System" Human Language Technologies as a Challenge. Proceedings of the 2nd Language and Technology Conference, Wydawnictwo Poznanskie.

[19] Z. Wang, A. Ittycheriah (2015) "FAQ-based Question Answering via Word Alignment," quesarXiv: 1507.02628v1 [cs.CL] 9 Jul 2015. Microsoft 2018.[Online]: https://www.qnamaker.ai/

[20] A. Xu, Z. Liu, Y. Guo, V. Sinha and R. Akkiraju (2017) "A New Chatbot for Customer Service on Social Media" CHI '17 Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, 3506-3510. https://doi.org/10.1145/3025453.3025496

[21] A. Deshpande, A. Shahane, D. Gadre, M. Deshpande and P. M. Joshi (2017) "A Survey of Various Chatbot Implementation Techniques" International Journal of Computer Engineering and Applications, Vol. 11.

[22] J. Hill, W. R. Ford and I. G. Farreras (2015) "Real Conversations with Artificial Intelligence: A comparison between Human–Human Online Conversations and Human–Chatbot Conversations" Computers in Human Behavior, Vol. 49, 245–250. https://doi.org/10.1016/j.chb.2015.02.026

[23] D. Harris and S. Harris (2012) "Digital Design and Computer Architecture" Elsevier book 2nd edition, 129.

[24] Andrew L. Maas and Awni Y. Hannun and Andrew Y. Ng (2013) "Rectifier Nonlinearities Improve Neural Network Acoustic Models".

[25] Vinod Nair and Geoffrey Hinton (2010) "Rectified linear units improve restricted Boltzmann machines".

[26] T. Dozat (2016) “Incorporating Nesterov Momentum into Adam” Stanford ICLR 2016 workshop submission.

[27] A. Galiano, A. Massaro, D. Barbuzzi, L. Pellicani, G. Birardi, B. Boussahel, F. De Carlo, V. Calati, G. Lofano, L. Maffei, M. Solazzo, V. Custodero, G. Frulli, E. Frulli, F. Mancini, L. D’Alessandro, F. Crudele (2016) “Machine to Machine (M2M) Open Data System for Business Intelligence in Products Massive Distribution oriented on Big Data,” (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 7 (3), 1332-1336. Github dataset .[Online]: https://github.com/DyrectaLab/faq-data .