applied sciences

MDPI

Article

Development of Dialogue Management System for Banking Services

Samir Rustamov [1,2,*], Aygul Bayramova [1,3] and Emin Alasgarov [1,4,*]

[1] School of Information Technologies and Engineering, ADA University, Ahmadbey Aghaoglu Str. 11, AZ1008 Baku, Azerbaijan; abayramova99@gwu.edu
[2] Institute of Control Systems, Bakhtiyar Vahabzadeh Str. 9, AZ1141 Baku, Azerbaijan
[3] School of Engineering and Applied Science, The George Washington University, 1918 F Street NW, Washington, DC 20052, USA
[4] Unibank CB OJSC, Rashid Behbudov Str. 55, AZ1014 Baku, Azerbaijan
* Correspondence: srustamov@ada.edu.az (S.R.); emin.alasgarov@unibank.az (E.A.)

Abstract: Rapid increase in conversational AI and user chat data lead to intensive development of dialogue management systems (DMS) for various industries. Yet, for low-resource languages, such as Azerbaijani, very little research has been conducted. The main purpose of this work is to experiment with various DMS pipeline set-ups to decide on the most appropriate natural language understanding and dialogue manager settings. In our project, we designed and evaluated different DMS pipelines with respect to the conversational text data obtained from one of the leading retail banks in Azerbaijan. In the work, the main two components of DMS—Natural language Understanding (NLU) and Dialogue Manager—have been investigated. In the first step of NLU, we utilized a language identification (LI) component for language detection. We investigated both built-in LI methods such as fastText and custom machine learning (ML) models trained on the domain-based dataset. The second step of the work was a comparison of the classic ML classifiers (logistic regression, neural networks, and SVM) and Dual Intent and Entity Transformer (DIET) architecture for user intention detection. In these experiments we used different combinations of feature extractors such as CountVectorizer, Term Frequency-Inverse Document Frequency (TF-IDF) Vectorizer, and word embeddings for both word and character n-gram based tokens. To extract important information from the text messages, Named Entity Extraction (NER) component was added to the pipeline. The best NER model was chosen among conditional random fields (CRF) tagger, deep neural networks (DNN), models and build in entity extraction component inside DIET architecture. Obtained entity tags fed to the Dialogue Management module as features. All NLU set-ups were followed by the Dialogue Management module that contains a Rule-based Policy to handle FAQs and chitchats as well as a Transformer Embedding Dialogue (TED) Policy to handle more complex and unexpected dialogue inputs. As a result, we suggest a DMS pipeline for a financial assistant, which is capable of identifying intentions, named entities, and a language of text followed by policies that allow generating a proper response (based on the designed dialogues) and suggesting the best next action.

Keywords: dialogue management systems; low-resource language; conversational artificial agent; intent classification; conversational chat-bot

Citation: Rustamov, S.; Bayramova, A.; Alasgarov, E. Development of Dialogue Management System for Banking Services. *Appl. Sci.* **2021**, *11*, 10995. https://doi.org/10.3390/app112210995

Academic Editor: Steven Walczak

Received: 21 September 2021
Accepted: 1 November 2021
Published: 19 November 2021

1. Introduction

The development of financial assistants equipped with a certain level of artificial intelligence tasks is one of the hot topics within the banking industry all around the world. Applied efforts mostly concentrate around two important directions: automation of bank operations by understanding users' actions and automation of client-operator conversations by understanding clients' intentions and problems. The former direction requires integration with internal (e.g., business process management system) and external systems, and leads to automation of many business processes (e.g., loan applications, card

orders, etc.). The latter direction relies on carefully designed conversations and user input processing and extremely reduces the load on call centers that receive a vast amount of repetitive questions. Both automation aims have to be backed up by solid natural language processing techniques that allow understanding of the domain language and generating a reasonable answer to user requests.

Financial assistants are considered a part of conversational software where the biggest challenge is to develop machine-learning based dialogue management systems (DMS) and support it with powerful language-understanding and language-generation modules. As a result, modern DMS architectures require several fundamental modules within the pipeline:

- Natural Language Understanding (NLU)—performs users' input processing and handles slot-values and sentence intention and entities;
- Dialogue Manager (DM)—maintains the dialogue state and suggests the next action to be taken or a sentence to be responded to;
- Natural Language Generator—receives a specific communication act from the DM and generates a natural language response.

In our work, natural responses are ready sentences that have been obtained from chat dialogues and are based on the NLU and DM steps being produced as a response during the conversation with the chat-bot. Therefore, we mostly focused on NLU and DM modules in the paper.

It should be noted that although there some studies, on topics such as speech recognition, speech synthesis, and sentiment analysis of texts [1–3], that have been conducted in the Azerbaijani language, the development of dialogue systems has been less studied. Furthermore, there is a lack of language processing tools for the Azerbaijani language. Some multilingual tools, such as BERT, fastText, and Spacy, trained on clean datasets from sources such as Wikipedia and text corpuses from books; thus, they are not effective for conversational chat data, which contain plenty of misspellings. Most of the misspelling issues are caused due to the alphabetical differences between the Azerbaijani and English languages and the usage of the English keyboard layout while typing in Azerbaijani. This issue is explained in more detail in Sections 3 and 4. Additionally, multilingual tools do not show good performance on our dataset, as their training data is not related to the bank domain. For this reason, we comparatively analyzed different machine learning methods and chose an effective set of techniques and parameters for our dataset. We anticipate that our experimental study will have a huge impact in our domain and will provide a good foundation for the upcoming work for the agglutinative low-resource languages.

In our project, we designed and evaluated different DMS pipelines with respect to the Azerbaijani conversational text data obtained from one of the leading retail banks in Azerbaijan, Unibank. However, semantics of the dataset, i.e., meanings of the Azerbaijani words, have not been considered in the research and did not affect data preparation steps. Used techniques, such as bag-of-words (BoW) representation, term frequency-inverse document frequency (TF-IDF) vectorization, word embeddings, machine learning (ML) algorithms, etc., can be applied to other languages as well. The results achieved using such techniques on the other languages also discussed in the work, but those datasets do not belong to the banking sector. In addition, since the Azerbaijani language belongs to the group of agglutinative languages, it is anticipated that these experiments will be effective in other agglutinative languages as well. We experimented with various combinations of NLU and DM modules to find the best suitable pipeline for our conversational dataset.

In Section 2, we discussed the major modules of the DMS pipeline by showing what kind of research has been already conducted within the domain.

Section 3 is dedicated to the explanation of the conversational dataset obtained from one of the leading retail banks of Azerbaijan, Unibank. There is given a general description of the data, the distribution of intentions, and difficulties of the language in the same section.

In Section 4, we analyzed major methods for building Language Identification, Intent Detection, Entity Extractor, and Dialogue Management modules. We investigated and

compared both built-in language identification methods, such as fastText, and custom machine learning models trained on a domain-based dataset. For intent detection we applied Dual Intent and Entity Transformer (DIET) and other traditional machine learning methods such as Naïve Bayes (NB), K-Nearest Neighbor (KNN), Logistic Regression (LR), Support Vector Machine (SVM), and Neural Networks (NN) with TF-IDF, CountVectorizer, sub-word embedding features to find the best model for our dataset. The NLU set-ups are compared and followed by a DM module that contains a rule-based Policy to handle FAQs and chitchats, as well as a transformer embedding dialogue (TED) policy to handle more complex and unexpected dialogue inputs.

In Section 5, we describe experimental results in detail. We conclude by discussing major findings and providing summary in Sections 5 and 6.

By the end of this work, we obtained a sophisticated DMS pipeline for a financial assistant, which is capable of identifying intentions, named entities, and language types, using proper policies that allows generating proper response (based on the designed dialogues), and suggesting the best next action for the chat-bot.

2. Related Work

In the article, our research mainly focused on Natural Language Understanding and Dialogue Management modules; therefore, related research in these areas is explored in the following paragraphs.

2.1. Natural Language Understanding Module

Standing at the frontline of a conversation with a user, the NLU module handles several aspects of natural language processing. First, sentences should be sliced into words (tokenized) and features should be extracted from those words to create embeddings. Obtained features are used by language identification, named entity recognition and intent classification components. A weak design of an NLU module can result in a disrupted dialogue state and an unpleasant conversational experience with the artificial assistant. We discuss recent achievements for the noted NLU components in the paragraphs below.

Tokenization and feature extraction are considered the most important components of any NLU module. The former method extracts features by converting text documents to a matrix of counted tokens, whereas the latter one takes an additional step and converts a count matrix to a normalized TF-IDF representation. Both approaches suffer computationally from sparse matrices, and for the last couple of years new methods have been proposed by the research community. Alternative to CountVectorizer and TF-IDF methods, word embedding methods such as Word2Vec, Glove, fastText, and others are generally used. Word2Vec [4] is a two-layer neural net used to create word embeddings with two different methods: Continuous Bag-of-Words (CBOW) and the Skip-Gram. CBOW uses context words to predict target words, whereas Skip-Gram uses target words to predict surrounding context words. GloVe [5] is considered to be a model with a weighted least-squares objective that incorporates the ratios of word-word co-occurrence probabilities as global statistics for vector encodings. FastText [6] represents each word as an n-gram of characters and it helps to understand suffixes and prefixes. Another modern and proven approach is BERT [7], which allows one to pre-train deep bidirectional representations from a text using both left and right contextual information for every word. BERT developers utilized an efficient combination of Mask Language Model and Next Sentence Prediction components to predict missing (expected) words in the text. In addition, this method deals with homonyms quite efficiently due to its bidirectional analysis of a given sentence. Word embeddings for the Azerbaijani language have been investigated in [8]; however, their dataset is taken from news and Wikipedia and is not efficient for our task.

Another important component of the NLU module is language identification, and we have evaluated several methods in order to define the best approach for Azerbaijani language detection in our task. Significant research has been conducted within the language identification domain [9]. Many scholars tried to tackle multilingual text documents in their

work [10–12]. Ranjan et al. proposed a novel approach based on a deep neural network where training of a model happens with respect to the very limited dataset of feature vectors [13]. The authors of [14] investigated performance of several machine-learning algorithms with the application to social media short messages using both probabilistic and non-probabilistic models. The idea of using short texts for language identification has been described by Perełkiewicz et al. in their work [15] where they used a Long Short-Term Memory (LSTM) Neural Network augmented with the attention mechanism. In our work, we experimented with fastText [6] developed by Facebook that contains pre-trained multi-lingual word vectors for 157 languages trained on Common Crawl and Wikipedia. We then compared its results to more traditional models trained via Naïve Bayes, Logistic Regression, Linear SVM, and KNN using our own dataset that contains distorted words of the Azerbaijani language.

The authors of [16] introduced Bidirectional Gated Recurrent Unit (BiGRU) where the current hidden state is used to tag an entity and the hidden state of the previous step is used to classify an intent. Another approach for user intention detection is Neuro-Fuzzy Inference systems in Call Routing Systems [17,18].

Certain research [19,20] has been conducted around attention based bidirectional Long Short-term Memory (BiLSTM) for intent classification and sequence labeling where the enriched encoder-decoder model is capable of mapping sequences with different lengths without any alignment information. Vanzo et al. introduced a hierarchical multi-task architecture made of the self-attention mechanism and BiLSTM encoders followed by conditional random field (CRF) tagging layers [21]. Researchers at RASA [22] have introduced DIET architecture that allows for incorporating pre-trained word embeddings obtained from language models and then combining them with character level n-grams and sparse words [23].

Most of the noted methods have been applied to the English language dataset, and it is difficult to predict how the same techniques can be successful for noisy conversational texts in the Azerbaijani language. For this purpose, we trained and analyzed different ML algorithms such as SVM, NB, LR, NNs, DIET, and KNN to get the best fitted method for our dataset.

2.2. Dialogue Management Module

The first generation of DMS is considered as rule-based [24] and flowchart-based that simulate state transitions during the dialogue that are based on the finite state automata [25,26]. Both systems require intense domain expert support and cannot be considered flexible with respect to unexpected changes in a dialogue flow.

The rapid increase in data has led to the second generation of statistical data based DMS, where the Partially Observable Markov Decision Process (POMDP) is considered as one of the robust representatives [27]. In this type of system, every dialogue stage possesses a state supported by a Bayesian inference, and a particular dialogue policy is selected in order to generate a proper natural language response. The authors of [28] have shown how the continuous component of POMDP can enable the model to incorporate a confidence score of the automated dialogue management and to outperform traditional Markov decision processes. Thomson et al. took another approach in their work [29], where they showed how the sequence of certain dialogues are used to estimate Q function to establish the dialogue state and the next action.

The third generation of DMS emerged recently after the breakthroughs in deep learning and the drastic increase in conversational data, being more defined as goal-oriented dialogue systems [30–33]. During this generation, there was a clear shift from generative models, such as Bayesian networks, to deeply discriminative ones, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) [25].

Korpusik et al. showed how to utilize CNN to generate semantic tagging of each utterance in order to update the dialogue state [34]. Wang et al. discussed how to use CNN with a maximum pooling layer to obtain local features from the provided utterances and

then select the most important semantic factor [35]. The authors of [36] propose a neural network framework made of two steps: supervised learning from dialogue data followed by the continuous improvement using the reinforcement learning.

Sidner and Groz in their work [37] suggest handling a dialogue state by assuming that a particular dialogue is a stack of topics. More recent work introduced by the authors of RavenClaw [38] elevated the idea of the stack and applied it for sub-dialogues. One of the major drawbacks of stack-based architectures is that their structure is strict, and minor deviations from topics within a stack result in a loss of the context and a mix of the dialogue states.

RNNs started to become more popular in recent years, and showed promising results within general, open domains in dialogue systems [39,40]. RNNs in their default settings consume the entire sequence of input elements for encoding purposes, which increases the size of a trained model significantly. Long short-term memory based RNN architectures have proven to perform efficiently when it comes to the natural response generation for goal-oriented dialogue management systems [41–43]. However, one of the drawbacks of RNNs is that for low-resource and limited corpus, such as the Azerbaijani language, they do not guarantee dialogue state generalization. LSTM type RNNs tend to distort their internal state if, during a dialogue stage, an unexpected sentence is provided (e.g., user starts with credit application and then suddenly switches to debit card order).

Recently, the transformer architecture started to substitute RNNs for the language model training [44–47]. One of the major advantages of transformers is that they achieve independency while making predictions on different dialogue stages based on the self-attention mechanism. This mechanism is used to learn sentence representations by comparing different positions of that sentence and allows preselecting tokens that affect the current state of the encoder [48,49]. The authors of [50] developed the Recurrent Embedding Dialogue Policy (REDP) architecture that utilizes the attention mechanism to achieve a better performance while recovering from unexpected dialogue inputs. In their work [48], Vlasov et al. simplified the architecture of REDP and introduced TED policy. TED policy tries to maximize a similarity function while jointly training embeddings for a dialogue state and system actions.

In the work, we implemented a hybrid model using both TED Policy to handle contextual conversations and Rule Based Policy to handle the specific cases, such as FAQs.

3. Dataset and Pre-Processing

Dataset used in this research is LiveChat data from the banking domain in the Azerbaijani language. It consists of 161,856 customer-to-operator chats. The amount of customer messages is 1,894,543 out of 3,178,989 total sentences. Although the dataset is related to the narrow field of the financial sector, due to the agglutinative nature of the Azerbaijani language and noise of the online chat data, it has large vocabulary content (450,321 unique words). For example, because of the misspellings, online chatting style, and usage of different languages there can be encountered several variations of the same word, such as *"kredit"*, *"credit"*, *"кредит"*, *"kridit"*, *"kiredit"*, etc., all referring to the word *"credit"*. Usage of different keyboard layouts between English and Azerbaijani can also cause other variations of the words and sentences. Let us consider following two sentences: "Mən onlayn kredit sifariş vermək istəyirəm" and "Men online kridit sifarish vermek isteyirem". In these examples, the first sentence consists of orthographically correct words, but in the second sentence all the words are misspelled. The contextual meanings of these two sentences are absolutely identical—"I would like to order a credit card", however we ended up with 12 unique word forms of 6 total words. Overall, in the used dataset, 74% of all sentences have misspelled words.

Another reason for having many different forms of words is the morphological structure of the agglutinative languages, e.g., having suffixes. For example, some of the morphological forms of the word "*kart*" ("*card*") are: "*kartı*", "*kartım*", "*kartımı*", "*karta*", "*kartıma*", "*kartımın*", "*kartın*", "*kartlar*", "*kartları*", etc.

A total of 293,182 words are used only once throughout the whole dataset. The quantity of words that were encountered more than five times is 59,861. Distributions of the frequencies of words are shown in the bar chart in Figure 1.

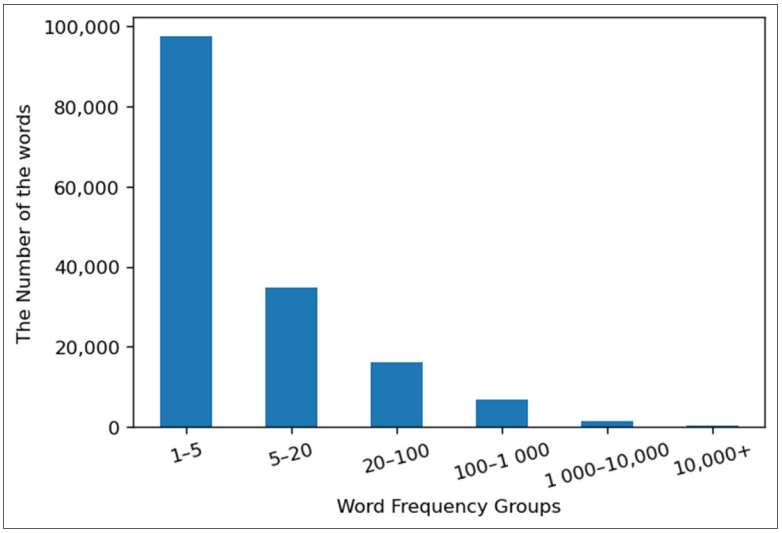

Figure 1. Word frequency group distribution.

In our work, the most recent 10,000 chats were processed, and their intent manually labeled by human agents. Intent refers to the goal of a customer during the conversation with an agent. Intent of a user message in the chat-bot can be a question or an operation that the user wants to execute. Various examples can be seen in the Table 1.

Table 1. Chat-bot intent examples.

Message	Intent
"Mən sizin bankda kredit sifariş etmək istəyirəm." [1] (*I would like to order a credit card in your bank*) [2]	credit request
"Debet kartı necəaktivləşdirəbilərəm?" (*How can I activate the debit card*?)	card activation
"Kredit borcumu ödəmək istəyirəm." (*I would like to pay my credit debt*.)	credit payment

[1] Original user message in Azerbaijani language; [2] English translation of the original text.

First user messages in each thread were used to detect an intent of a conversation. Other sentences were used in conversations to handle a dialog flow or added as chitchat and FAQ questions. Additionally, messages of greetings and gratitude were added to the intent set, which together formed a total of 63 intents. On average, for each intent the number of examples is 161, and the maximum number of examples encountered for all the types of "card request" is 1589. Information about distribution of intents is represented in Table 2 and Figure 2.

Table 2. Intent distribution.

	Intent	Number of Samples
1	card_request	1589
2	credit_info	1215
3	unblock	434
4	limit_increase	328
5	work_time	312
.	.	.
.	.	.
.	.	.
59	other	21
60	chitchat/ask_weather	17
61	stop_credit_application	16
62	ubank_operations	15
63	chitchat/url_not_working	10

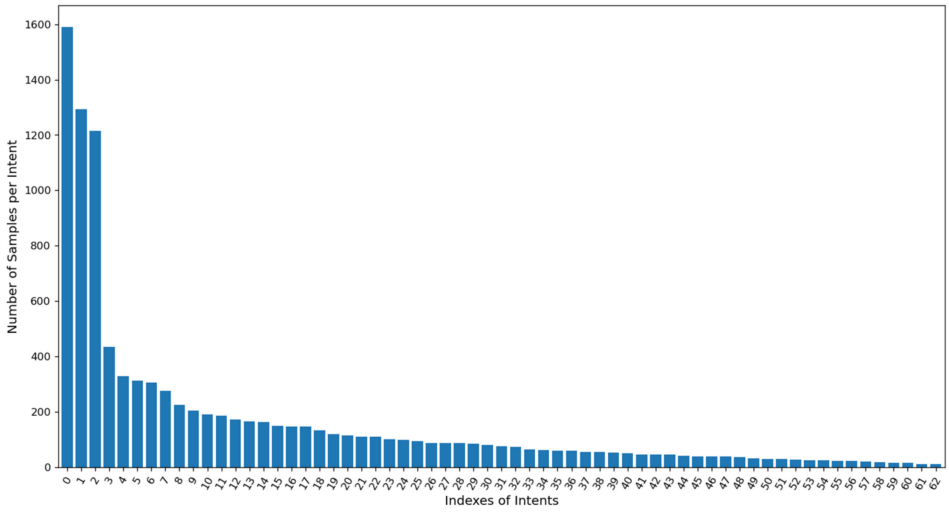

Figure 2. Distribution of all intents throughout the dataset.

4. Methods

Modular and End-to-End modeling are two main approaches in conversational chat-bot development [23]. Whereas Modular systems consist of two separate modules, user intention understanding and dialog management, End-to-End systems learn dialog patterns directly from interactive human-to-human conversations. In this research, we focused on a Module-based conversational agent, which consists of three main modules (Figure 3):

- *NLU Module.* The goal of the NLU module is to process user messages, understand intent, and extract named entities. Intent identification means defining the main purpose of the user in a message. Named entity extraction is used to get important entities from text, such as person names, organizations, geological places, dates, time, etc.
- *DM Module.* Dialogue management is a machine learning based system that predicts the next bot action or response according to the previous dialogue turn, i.e., user-bot interactions, extracted from intent and entities.
- *Natural Language Generation (NLG) Module.* Currently, the NLG module of the bot consists of preformed sentences, which can be used as bot responses.

Additionally, the chat-bot has an Integration Module to connect to the other environments. The integration module plays a main role in executing the operational part of the chat-bot, such as checking the user's account and integrating with internal and external services.

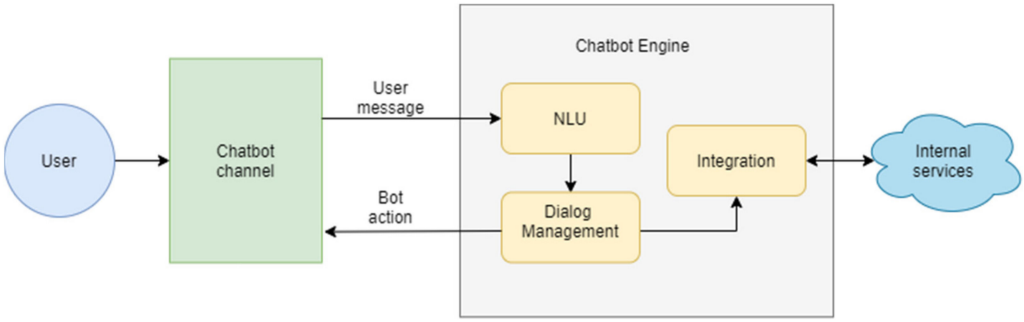

Figure 3. Architecture of the module-based chat-bot system.

The NLU component itself consists of several steps, such as text representation (tokenization and feature extraction), language identification, and user intent detection. All these steps can be seen in Figure 4.

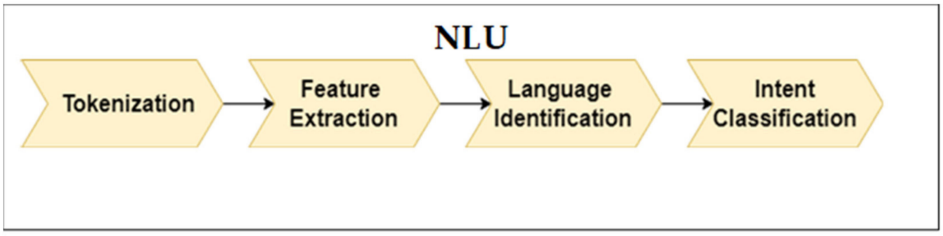

Figure 4. Part of natural language understanding (NLU) component.

4.1. Tokenization and Feature Extraction

The dataset introduced earlier has specific characteristics, such as misspellings and examples with an incomplete sentence structure. One of the main reasons of having many misspellings in Azerbaijani chat data is alphabetical differences between Azerbaijani and English. They both are Latin-script alphabets, but in the English alphabet there are 26 letters, whereas in the Azerbaijani alphabet there are 32. The Azerbaijani letters—"ç", "ş", "ğ", "ə", "ü", "ö" do not exist in the English alphabet, and the letter "w" does not exist in the Azerbaijani alphabet. Thus, most of the misspellings occur due to the usage of the English keyboard layout. For example, instead of "ə" and "ş", customers may use "a", "e", and "sh", "s", "w" correspondingly, resulting in several wrong variations of the same word (e.g., correct word: "şəhər"—misspellings: "sheher", "sahar", "weher", etc.). Other types of misspellings that occur in the dataset are caused by missing or adding a letter, confusing or switching the order of the letters, etc. Considering the amount of the data and the problems explained above, whitespace tokenizer and sparse features are implemented in the data processing step. Thus, tokens obtained using whitespace as a separator fed into the count vectors featurizer, which results in a BoW representation. BoW representation, known as one-hot encoding, creates sparse features for a given text dataset. In other words, the resulting representation consists of unique vectors with zeros and ones for each word with the dimension that is equal to the vocabulary size. In order to extract the internal features of words, the character n-gram model is applied with parameters min_ngram = 2 and max_ngram = 4. In other words, we implemented both word level and character n-gram level BoW representations of the dataset. Additionally, character n-gram representations are effective in handling misspelling problems encountered in the dataset.

Apart from this, for machine learning models TF-IDF Vectorizer is also used. Unlike the BoW method, this creates occurrence-based representations. Here, a greater importance is given to the words that have more occurrences in the given text. In contrast, less importance is given to words that are common across all intent classes, and their impact in intent identification is very low.

In addition, named entities extracted from user messages were added as features. The main goal of the named entity is to recognize important information from the raw text, for example, person names, organizations, geological places, dates, time, etc. In conversational AI, named entity recognition (NER) can be implemented to extract general entities such as those listed above, as well as domain based custom entities that are helpful to understand customer needs and respond accordingly. For instance, in the financial industry, especially within the bank domain, information like type of a credit or deposit, plastic card differences, interchange fee, and transaction percentage is considered valuable. Along with Part of Speech (PoS) tagging, NER related to the sequence labeling problems of NLP is important part of Information Extraction (IE) and Question Answering (QA) [51,52]. For NER classification, different machine learning methods can be implemented. One of the approaches for entity recognition is using CRF. CRF can be implemented inside DIET Classifier, if the class of entity recognition is set. In this case, Entity Loss and Intent Loss are summed up to get a total loss of DIET Classifier. Both sparse and dense features first go through two transformer layers, followed by the CRF tagging layer for entity label prediction.

Apart from this, CRF based entity recognition can be implemented separately by using custom entity extraction models. In our work, we implemented the CRF entity extraction using different tagging methods to handle the multi-word entity problem, for example "credit card". One of the methods is known as Beginning Inside Last Out Unit (BILOU) tagging [23,52]. BILOU tagging is mostly used to label multi-word entities. For example, we have a "albali credit card" which is the "card_type" entity. To label this entity in our training data we use [albali](b-card_type) [credit](i-card_type) [card](l-card_type). Here b-, i-, l- are prefixes meaning beginning, inside, and last, accordingly. Additionally, u- (unit, for example [ubank](u-app)) tag is used to label one-word entities, and o-(out) is used to label words that are not the entities. The second method is using the part(particle) tag to note the word being the part of a multi-word entity. For instance, the entity example above will be tagged as [albali](part) [credit](part) [card](card type). Another technique applied to handle multi-word entities is training sentence samples, which include multi-word entities separately from the sentences that include only one-word entities.

We also applied NN for NER classification. However, performance of the NN were similar or lower than CRF for our dataset. For this reason, we proceeded further with only the CRF entity extractor.

4.2. Language Identification

Another problem that occurred during the development was handling messages in different languages used by the bank customers that are not fully supported by the NLU model of the chat-bot. The main reason for having different languages in our dataset is that certain groups of people in Azerbaijan speak Russian or English. For this reason, we get about 3% of the inquiries in non-Azerbaijani, most being in Russian, English, Turkish, etc. To tackle listed problems, a Language Identification component has been added to the bot model. Two main approaches can be implemented to identify the language of the message: applying existing pre-trained language models and developing custom language identifiers.

First, the available approaches that support the Azerbaijani language, such as Facebook's "fastText" language identification model and "langid" Python package for language identification, were investigated. The "langid" package is trained over Naive Bayes Classifier using a mixture of byte n-grams. The package itself comes together with an embedded model, which supports 97 languages and is trained using the data extracted from variety of domains, in order to handle diverse types of textual data [53]. However, "fastText" text-based language identification model can identify 176 languages, including Azerbaijani [6]. Using both BoW and character n-grams approaches in the model architecture, as well as implementing hierarchical softmax, "fastText" achieves high performance and reduces computation time. For our work, "fastText" language identification has been utilized. The

"fastText" model has two language models: the light model with the file size of 917 kB and the heavy model with the file size of 126 MB, which is considered to be more accurate. We implemented language identification using both models. Both models were pre-trained on a clean dataset from Wikipedia, Tatoeba, and SETimes, and the results were not accurate with respect to our noisy online chat dataset.

Second, by implementing machine learning algorithms on the domain dataset, custom methods were established. For this purpose, we used Naïve Bayes, Logistic Regression, Linear SVM, and KNN classifiers to train a two-class language identification model. These models are used to distinguish the Azerbaijani language from other languages, primarily English and Russian.

4.3. Intent Classification

The next step of the NLP based chat-bot development is intent classification. Intent classification helps to identify customer intention and detect needs from the provided text message. For intent classification, we have applied several classic machine learning methods and the self-attention mechanism-based DIET Classifier.

The DIET Classifier is an attention based dual intent and has entity transformer architecture [23]. The main advantage of the DIET architecture is its capability of processing words, character n-gram level sparse features, and pre-trained word embeddings at the same time. Both intent classification and entity recognition problems can be addressed inside the DIET Classifier. The architecture has three components: intent classification, entity recognition, and masked language model. Total loss of the model can be calculated with:

$$L_{total} = L_I + L_E + L_M$$

where L_I is intent classification loss, L_E is entity recognition loss, and L_M is masked language loss.

Additionally, custom intent classification models were developed using classic machine learning techniques. In our work, we utilized and tuned NB, SVM, KNN, logistic regression, and neural networks.

The NB classifier is a statistical learning algorithm based on Bayes theorem. The prediction of a sentence belonging to a certain category using the Naïve Bayes method is based on the probability assigned to each term. These probabilities are found according to the occurrences of a term in the training data. The sum of the probabilities of all tokens is then calculated to detect the category of the text.

SVM is a linear classification method that is currently considered to be stable. The objective of the support vector machine algorithm is to find a hyperplane in N-dimensional space (N—the number of features) that distinctly classifies the data points. To classify unknown points, it will be enough to look at which side of the hyper-plane they will be located.

Moreover, KNN, logistic regression algorithm, and neural networks architecture were applied for the intent classification. The results of the Naïve Bayes and KNN models have shown low performance for our dataset. Thus, we focused on DIET Classifier, SVM, LR, and NNs approaches.

4.4. Dialogue Management

Artificial assistants (i.e., AI chat-bots) can be classified into three major categories based on the pursued goals [25,54]: informative chat-bots, task-based chat-bots, and conversational chat-bots. Informative chat-bots answer user questions according to the fixed source. A great example of this category is FAQ chat-bots. Task-based chat-bots perform different operations, such as checking user accounts, room booking, etc. Conversational chat-bots are known to be more interactive, can hold human-like conversations and have context switches.

The Dialog Management module controls and updates the conversation context. After intent classification, DM predicts the next best action that should be performed by a

bot (Figure 5). Dialog management modules of conversational chat-bot systems can use rule-based, machine learning based, or hybrid models. Rule-based systems follow strict conversational behavior, which arises from the rules defined in the training data. It matches a user message to a predefined pattern and responds with a fixed answer. Whereas in rule-based chat-bots the next bot action depends on the current turn, in machine learning based systems, the bot action is predicted by considering the whole dialog context, several previous dialogue turns, extracted entities, and filled slots. They have the ability to learn from conversation but require a large dataset to train the ML model. A hybrid system uses a mixture of both techniques and is considered to be more efficient.

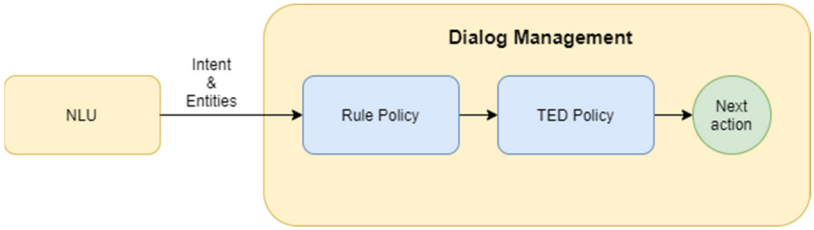

Figure 5. Dialog manager (DM) module.

In our work, both rule-based and machine learning-based methods are applied for dialogue management. Rules are used to handle conversations with chitchat and FAQ questions to return a unique answer for each situation. In certain scenarios, the system cannot identify an intent based on the certain prediction threshold, which is referred to as a fallback. Fallbacks can also rise due to the absence of a specific response for a provided intent. On the other hand, rule-based approaches provide options for the cases when specific information should be received from users, such as a name, birth date, etc. Architecture of the DM system is represented in Figure 5.

TED policy is one of the efficient machine learning based dialog management methods [48]. TED policy takes an intent and entities from the user input, the slot and form values, and bot actions from previous turns as features, feeds them into the transformer where the self-attention mechanism is applied. Outputs of the transformer goes through the embedding layer and similarity between resulting embedding and the actual bot action is calculated to determine the loss value of one step.

The advantage of the TED policy is that it can be used in both modular and end-to-end chat-bot systems. When using TED policy in end-to-end systems, instead of having fixed labels for NLU outputs and bot actions, user messages and bot responses are represented as a BoW vectors. In this work, a hybrid model of dialog management has been implemented by using both the rule-based approach and TED policy. Rule policy is applied to the cases in which the user message with a specific intent has a fixed answer, such as FAQ and chitchats. In contrast, TED policy is used to handle complex and contextual conversational patterns where the user-bot interaction is needed.

4.5. Integration Module

Handling natural language conversations is as important as making sure that business goals declared for an artificial assistant are achieved properly. In that regard, the results of NLU and DM modules serve as an input for the Integration Module of our chat-bot. One of the major components inside the Integration Module is the action server that consumes requests from the DM to execute a particular custom action (i.e., return user balance or start online credit application). The action server can be integrated with the internal and, external systems through simple API calls to accomplish required business processes. Among internal systems, we have integrated our chat-bot with the Business Process Management (BPM) system of Unibank. The BPM system serves as a business logic backend and achieves a good level of abstraction hiding the development of all complex internal services. We helped to automate several business processes, including the online credit application, plastic card order (credit or debit), user balance, and transaction history

display. In addition, our system is integrated with external chat systems, and currently, we communicate with clients through LiveChat and Telegram. LiveChat is installed at the original web-page of Unibank and serves as a conversational interface that provides questions and answers based on the developed NLU and DM modules. The test Telegram channel of Unibank provides more flexibility and, for certain business cases, communicates with the BPM system to accomplish internal computations and automate bank services. Conversations that pass through Telegram are also handled by NLU and DM modules in order to detect the next best action or response.

Integration of the chat-bot to the other banking services and the user interface consists of several steps. First, all the training data and model pipeline configuration files are uploaded to the remote repository to be used by the test and development servers. The Continuous Integration/Continuous Development pipeline is then built to automate training, testing, and deployment processes. Trained NLU and DM models transfer to the bot server where it is connected to the BPM to get customer details and information about transaction and to the messenger services, such as Telegram or LiveChat (Figure 6).

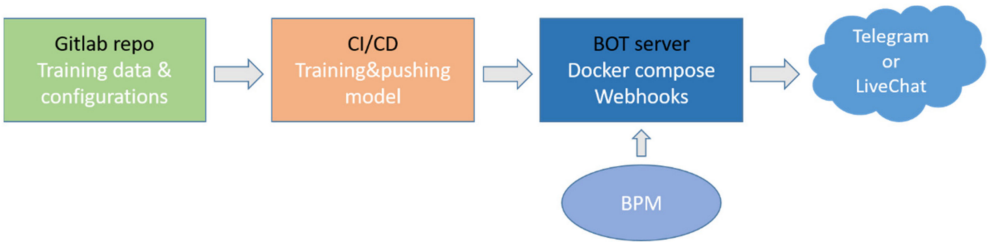

Figure 6. A simplified schematic representation of the Integration Module.

5. Results

As evaluation metrics, we have used accuracy, precision, recall, and F1-score in our system for classification. Accuracy is the ratio of correctly predicted instances to the total instances. Precision is the ratio of correctly predicted positive instances to the total predicted positive instances. Recall is the ratio of correctly predicted positive instances to all instances in actual class. F1-score is the weighted average of precision and recall.

5.1. Entity Recognition

Out of the methods we implemented with CRF, results from BILOU tagging and particle tagging methods have similar accuracies at approximately 95%. In contrast, using separate models for one-word and multi-word entities provided better results, especially for the one-word model. All of the outcomes can be seen in Table 3.

Table 3. Named entity recognition (NER) results with various implementations.

Method	Accuracy	Precision	Recall	F1-Score
CRF + BILOU tagging	0.949	0.941	0.929	0.934
CRF + particle tagging	0.949	0.949	0.953	0.950
CRF + one-word model	0.983	0.981	0.983	0.981
CRF + multi-word model	0.955	0.952	0.956	0.952
DIET entity extraction	0.981	0.980	0.981	0.980

5.2. Language Identification

We implemented LI using the fastText language recognition component and classic ML algorithms. The results of both approaches are presented separately.

5.2.1. fastText Language Identification

The fastText language identification models are distributed in two versions, which were trained on the data from Wikipedia, Tatoeba, and SETimes. Although the heavy model (file size of 126 MB) is considered to be more accurate than the light model (file size

of 917 kB) for identifying 176 languages, using fastText models on our dataset, we achieved 70% and 71% total accuracies with the heavy and light models, respectively (Table 4).

Table 4. Language identification results using pre-trained fastText models. Reported numbers for precision, recall, and f1-score are weighted average scores.

Model	Accuracy	Precision	Recall	F1-Score
fastText Heavy Model	0.709	0.708	0.709	0.708
fastText Light Model	0.715	0.711	0.715	0.713

We achieved better results in identifying the Azerbaijani language with the heavy model, but correct predictions of "non_az" class were higher with the light model. (Table 5).

Table 5. fastText language identification accuracy results for separate classes.

Result	Light Model	Heavy Model
Total correct predictions	0.715	0.709
Total correct 'az' predictions	0.546	0.553
Total correct 'non_az' predictions	0.797	0.785

5.2.2. Language Identification Using Custom ML Models

Results of the custom models trained on the bank domain dataset are higher than fastText language identification. Accuracy of the classic ML model is 98–99% for different models. Detailed representation of the results can be found in Table 6.

Table 6. Results of the machine learning methods trained on over the domain dataset. Reported numbers for precision, recall, and F1-score are weighted average scores.

Model	Accuracy	Precision	Recall	F1-Score
K-Nearest Neighbors	0.98	0.98	0.98	0.98
Support Vector Machines	0.99	0.99	0.99	0.99
Logistic Regression	0.99	0.99	0.99	0.99
Naïve Bayes	0.98	0.98	0.98	0.98

5.3. Intent Classification

Results of the Naïve Bayes classifier and KNN methods were below 70%, therefore, they were not implemented in this research. In contrast, the performances of other classifiers change in the range of 79–89%. Four different ML algorithms (LR, SVM, NNs, and DIET) with nine various features (CountVec Word, CountVec ngram, TF-IDF word, TF-IDF ngram, TF word, TF ngram, fastText TF-IDF + ngram, fastText + ngram, fastText ngram + CountVec) were trained with 36 different models and the results are given in Tables 7 and 8.

Table 7. Intent classification results with sparse features. Reported numbers are accuracy and weighted average F1-scores.

	CountVectorizer Word		CountVectorizer ngram		TF-IDF Word		TF-IDF ngram		TF Word		TF ngram	
	Acc.	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
LR	0.79	0.80	0.87	0.87	0.83	0.82	0.88	0.88	0.82	0.83	0.88	0.88
SVM	0.83	0.83	0.88	0.88	0.85	0.84	0.89	0.88	0.84	0.83	0.89	0.88
NN	0.79	0.80	0.86	0.86	0.81	0.83	0.87	0.88	0.81	0.82	0.87	0.88
DIET	0.79	0.79	**0.89**	**0.89**	0.81	0.81	0.88	0.88	0.80	0.80	0.88	0.87

Table 8. Intent classification results with fastText embeddings. Reported numbers are accuracy and weighted average F1-scores.

	fastText (ngram-2-5) + TF-IDF		fastText (ngram-2-5)		fastText (ngram-2-5) + CountVectorizer	
	Acc.	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
LR	0.85	0.84	0.84	0.84	0.89	0.88
SVM	0.84	0.83	0.82	0.81	0.89	0.88
NN	0.87	0.87	0.82	0.83	0.88	0.89
DIET	0.868	0.866	0.878	0.876	**0.882**	0.879

5.4. Dialogue Management

The DM module of the bot is trained with 136 real conversation blocks using a hybrid model, which consists of TED policy and the rule-based policy. The training dataset for the DM part follows the pattern "user_intent—bot_action" turns. Here "user_intent" indicates the name of the intent class and "bot_action" is the name of the bot response. However, the test dataset is in "user_message—bot_action" format, where "user_message" is real textual data. The test dataset contains overall 94 conversation blocks with real test cases. It consists of 234 user messages and 276 bot responses. Although the accuracy of the correct predictions of the bot actions was higher than 83%, only 50% of the test dialogues were completely correct until the end. Detailed information about the results can be seen in Table 9.

Table 9. Dialogue Manager (DM) test results. Reported numbers are accuracy and weighted average F1-scores, precision and recall.

	Accuracy	**F1-Score**	**Precision**	**Recall**	**Total Correct Conversations**
Results	0.83	0.84	0.89	0.81	0.50

Overall, finalized entity recognition, language identification, and intent classification parts were utilized to form an NLU module of the bot. The NLU module is the part of the chat-bot that is used to understand user messages and extract important information. This model reduces the burden on the financial assistant and ensures that customer inquiries are processed quickly. Additionally, most of the tasks that were done by human agents can be automated after understanding customer queries. Thus, the tasks processed manually by human agents and time spent to queue up with call centers or online consultations are also decreased significantly.

6. Discussion

Main outcomes of this research can be divided into four groups: NER classification, language identification, intent classification, and dialog management. We used named entities for two purposes: as features for the intent classification and for the dialog management. In the dialog management part of the bot, achieving the highest accuracy for entity extraction is considered as one of the main goals because responses of the bot in different dialog turns highly depend on the extracted entities. In the experiments, we achieved accuracy for CRF models at approximately 95%, and DIET Classifier and the hybrid models for multi-word and single word entities showed the highest performances at about 98%. To increase the accuracy of the overall entity extraction model, we applied the hybrid model using the DIET Classifier and the multi-word entity extraction model. Authors in [21] suggested HERMIT architecture for detecting named entities in the NLU-Benchmark dataset for the English language and obtained approximately an 85% F1-score. Similar results (approximately 86%) were achieved by using DIET architecture in [23]. Considering the differences in utilized datasets, various language characteristics, and the number of entity classes, comparing our results with the research benchmark was problematic to a certain degree.

Results of the language identification method represented by the authors of [10] were 96% with a model trained on the multilingual Wikipedia dataset. Additionally, probabilistic models with evidence accumulation outperformed other methods on a multilingual Twitter dataset [11]. Although fastText language identification models showed high accuracy at 92–99% on Wikipedia, TCL, and EuroGov data, the performance of both fastText models (light and heavy) were approximately 55% applied to the text samples with spelling mistakes and 89–94% on spelling-corrected samples (Azerbaijani text data). The reason for them not performing well is that the fastText models are trained on the clean data resources with correct orthography and sentence structure. In contrast, custom ML models trained on our dataset have 98–99% accuracy for different models. To conclude, due to higher accuracy (99%) and lower complexity compared to others, we added a Language Identification component trained with Logistic Regression approach to the final chat-bot model.

Despite the results of the DIET Classifier using only sparse features, it achieved 88% accuracy, and using both sparse and dense (ConveRT) features is achieved 90% on the NLU-Benchmark dataset, accuracy of DIET on our dataset changes in the range of 81–89%. It can be noted that for a noisy dataset in the agglutinative language, classic ML models such as SVM and NN almost show the same performances as the transformer base algorithm. The only work on intent classification problem in the Azerbaijani language were in [17,18]. They applied Neuro-Fuzzy methods and achieved approximately 90% accuracy for the dataset with four intention classes; however, in our task, the number of intent classes is 63.

In the Dialog Management part of the bot, a hybrid model is implemented using both TED policy to handle contextual conversations and rule based policy to handle specific cases, such as FAQs. Accuracy of the correct prediction of the bot actions is approximately 83%, and 50% of the test dialogues are completely correct until the end.

7. Conclusions

In the work, the main two components of dialogue management systems—NLU and Dialogue Manager—have been investigated. In the first step of NLU, we utilized the Language Identification component for language detection. We investigated both built-in LI methods such as fastText and custom ML models trained on the domain-based dataset. Experiments show that the accuracies of the custom models trained on the bank domain dataset are much higher (98–99%) than fastText language identification. The second step of the work was the comparison of the classic ML classifiers (logistic regression, neural networks, and SVM) and DIET architecture for the user intention detection. Four different ML algorithms (LR, SVM, NN, and DIET) with nine various features (CountVec Word, CountVec ngram, TF-IDF word, TF-IDF ngram, TF word, TF ngram, fastText TF-IDF + ngram, fastText + ngram, fastText ngram + CountVec) are trained with 36 different models. It was found out that the transformer-based algorithms with CountVectorizer + ngram features show the best performance (89%). The Dialogue Management module trained by transformer embedding dialogue policy shows 83% accurate next action prediction and 50% full conversation prediction.

Future research is expected to be performed on a larger dataset with different input features and model selections. Adding PoS tagging and spell correction components for feature extraction, the development of a multi-language chat-bot system by applying language identification to recognize more languages could be considered as potential future work for us.

Author Contributions: Conceptualization, S.R. and E.A.; methodology S.R. and A.B.; software, A.B. and S.R.; validation, A.B., E.A. and S.R.; formal analysis, A.B.; investigation, S.R.; resources, E.A.; data curation, A.B.; writing—original draft preparation, E.A., A.B. and S.R.; writing—review and editing, S.R. and E.A.; visualization, A.B.; supervision, S.R.; project administration, E.A.; funding acquisition, E.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Due to sensitive information about bank customers, research data will remain confidential.

Acknowledgments: This work was carried out in Research Laboratory at Unibank. We would like to thank Agasalim Mammadov, Nurana Hasanbayova, Ali Nasrullayev, Emil Alasgarov, and Igbal Huseynov for helping during the data preparation and research process.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Valizada, A.; Akhundova, N.; Rustamov, S. Development of Speech Recognition Systems in Emergency Call Centers. *Symmetry* **2021**, *13*, 634. [CrossRef]
2. Valizada, A.; Jafarova, S.; Sultanov, E.; Rustamov, S. Development and Evaluation of Speech Synthesis System Based on Deep Learning Models. *Symmetry* **2021**, *13*, 819. [CrossRef]
3. Rustamov, S. A Hybrid System for Subjectivity Analysis. *Adv. Fuzzy Syst.* **2018**, *2018*, 2371621. [CrossRef]
4. Mikolov, T.; Chen, K.; Carrado, G.; Dean, J. *Efficient Estimation of Word Representations in Vector Space*, 1st ed. [ebook]. 2013. Available online: http://arxiv.org/pdf/1301.3781.pdf (accessed on 21 June 2021).
5. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*; Association for Computational Linguistics: Doha, Qatar, 2014; pp. 1532–1543.
6. fastText–Library for Efficient Text Classification and Representation Learning. 2014. Available online: https://fasttext.cc/ (accessed on 29 June 2021).
7. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; pp. 4171–4186. [CrossRef]
8. Huseynov, K.; Suleymanov, U.; Rustamov, S.; Huseynov, J. Training and Evaluation of Word Embedding Models for Azerbaijani Language. In *Digital Interaction and Machine Intelligence (MIDI 2020). Advances in Intelligent Systems and Computing*; Springer: Cham, Switzerland, 2021; Volume 1376. [CrossRef]
9. Jauhiainen, T.; Lui, M.; Zampieri, M.; Baldwin, T.; Lindén, K. Automatic Language Identification in Texts: A Survey. *J. Artif. Intell. Res.* **2018**, *65*, 675–782. [CrossRef]
10. Lui, M.; Lau, J.H.; Baldwin, T. Automatic Detection and Language Identification of Multilingual Documents. *Trans. Assoc. Comput. Linguist.* **2014**, *2*, 27–40. [CrossRef]
11. Nguyen, D.; Dogruöz, A.S. Word Level Language Identification in Online Multilingual Communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Seattle, WA, USA, 2013.
12. Li, L.; Yu, S.; Zhong, L.; Li, X. Multilingual Text Detection with Nonlinear Neural Network. *Math. Probl. Eng.* **2015**, *2015*, 431608. [CrossRef]
13. Ranjan, S.; Yu, C.; Zhang, C.; Kelly, F.; Hansen, J. Language recognition using deep neural networks with very limited training data. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016. [CrossRef]
14. Balazevic, I.; Braun, M.; Müller, K.-R. Language Detection for Short Text Messages in Social Media. *arXiv* **2016**, arXiv:1608.08515.
15. Perełkiewicz, M.; Poświata, R. Text Language Identification Using Attention-Based Recurrent Neural Networks. In *Artificial Intelligence and Soft Computing (ICAISC 2019). Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2019; Volume 11508. [CrossRef]
16. Zhang, X.; Wang, H. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*; AAAI Press: Palo Alto, CA, USA, 2016; pp. 2993–2999.
17. Aida-zade, K.; Rustamov, S. Learning User Intentions in Natural Language Call Routing Systems. In *Recent Developments and New Direction in Soft-Computing Foundations and Applications*; Springer: Cham, Switzerland, 2016; pp. 37–46.
18. Rustamov, S.; Mustafayev, E.; Clements, M.A. Context Analysis of Customer Requests using a Hybrid Adaptive Neuro Fuzzy Inference System and Hidden Markov Models in the Natural Language Call Routing Problem. *Open Eng.* **2018**, *8*, 61–68. [CrossRef]
19. Liu, B.; Lane, I. Attention-based recurrent neural network models for joint intent detection and slot filling. In Proceedings of the Interspeech 2016, San Francisco, CA, USA, 8–12 September 2016; pp. 685–689. [CrossRef]
20. Varghese, A.S.; Sarang, S.; Yadav, V.; Karotra, B.; Gandhi, N. Bidirectional lstm joint model for intent classification and named entity recognition in natural language understanding. In *Intelligent Systems Design and Applications*; Springer International Publishing: Cham, Switzerland, 2020; pp. 58–68.
21. Vanzo, A.; Bastianelli, E.; Lemon, O. Hierarchical multi-task natural language understanding for cross-domain conversational ai: Hermit nlu. *arXiv* **2019**, arXiv:1910.00912.

22. Bocklisch, T.; Faulkner, J.; Pawlowski, N.; Nichol, A. Rasa: Open Source Language Understanding and Dialogue Management. *arXiv* **2017**, arXiv:1712.05181.
23. Bunk, T.; Varshneya, D.; Vlasov, V.; Nichol, A. DIET: Lightweight Language Understanding for Dialogue Systems. *arXiv* **2020**, arXiv:2004.09936.
24. Webb, N. Rule-Based Dialogue Management Systems. In Proceedings of the 3rd International Workshop on Human-Computer Conversation, Bellagio, Italy, 3–5 July 2000.
25. Progress in Dialog Management Model Research. Available online: https://www.alibabacloud.com/blog/progress-in-dialog-management-model-research_596140 (accessed on 19 June 2021).
26. Finch, J.D.; Choi, J.D. Emora STDM: A Versatile Framework for Innovative Dialogue System Development. In Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 1st Virtual Meeting, 1–3 July 2020; pp. 261–264.
27. Young, S.; Gašić, M.; Thomson, B.; Williams, J.D. Pomdp-based statistical spoken dialog systems: A review. *Proc. IEEE* **2013**, *101*, 1160–1179. [CrossRef]
28. Williams, J.D.; Poupart, P.; Young, S. Partially Observable Markov Decision Processes with Continuous Observations for Dialogue Management. In *Recent Trends in Discourse and Dialogue. Text, Speech and Language Technology*; Dybkjær, L., Minker, W., Eds.; Springer: Dordrecht, The Netherlands, 2008; Volume 39. [CrossRef]
29. Thomson, B.; Schatzmann, J.; Weilhammer, K.; Ye, H.; Young, S. Training a real-world POMDP-based dialogue system. In *Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies (NAACL-HLT-Dialog '07)*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2007; pp. 9–16.
30. Sahay, S.; Kumar, S.H.; Okur, E.; Syed, H.; Nachman, L. Modeling Intent, Dialog Policies and Response Adaptation for Goal-Oriented Interactions. In *Proceedings of the 23rd Workshop on the Semantics and Pragmatics of Dialogue*; SEMDIAL: London, UK, 2019.
31. Ilievski, V.; Musat, C.; Hossmann, A.; Baeriswyl, M. Goal-Oriented Chatbot Dialog Management Bootstrapping with Transfer Learning. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI 2018), Stockholm, Sweden, 13–19 July 2018; pp. 4115–4121. [CrossRef]
32. Muise, C.; Chakraborti, T.; Agarwal, S.; Bajgar, O.; Chaudhary, A.; Lastras-Montano, L.A.; Ondrej, J.; Vodolán, M.; Wiecha, C. Planning for Goal-Oriented Dialogue Systems. *J. Artif. Intell. Res.* **2019**, *1*, 1–42.
33. Schaub, L.P.; Vaudapiviz, C. Goal-oriented dialog systems and Memory: An overview. In Proceedings of the 9th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics, Poznan, Poland, 17–19 May 2019.
34. Korpusik, M.; Glass, J. Dialogue State Tracking with Convolutional Semantic Taggers. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 7220–7224. [CrossRef]
35. Wang, Y.; Huang, J.; He, T.; Tu, X. Dialogue intent classification with character-CNN-BGRU networks. *Multimed. Tools Appl.* **2020**, *79*, 4553–4572. [CrossRef]
36. Su, P.-H.; Gasic, M.; Mrksic, N.; Rojas-Barahona, L.; Ultes, S.; Vandyke, D.; Wen, T.H.; Young, S. Continuously Learning Neural Dialogue Management. *arXiv* **2016**, arXiv:1606.02689.
37. Grosz, B.J.; Sidner, C.L. Attention, intentions, and the structure of discourse. *Comput. Linguist.* **1986**, *12*, 175–204.
38. Bohus, D.; Rudnicky, A.I. The ravenclaw dialog management framework: Architecture and systems. *Comput. Speech Lang.* **2009**, *23*, 332–361. [CrossRef]
39. Serban, I.V.; Sordoni, A.; Bengio, Y.; Courville, A.; Pineau, J. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*; AAAI Press: Palo Alto, CA, USA, 2016.
40. Henderson, M.; Thomson, B.; Young, S.J. Word-based Dialog State Tracking with Recurrent Neural Networks. In Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL); Association for Computational Linguistics: Philadelphia, PA, USA, 2014; pp. 292–299. [CrossRef]
41. Wen, T.H.; Gasic, M.; Mrksic, N.; Su, P.-H.; Vandyke, D.; Young, S. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Lisbon, Portugal, 2015. [CrossRef]
42. Tran, V.-K.; Le-Minh, N. Natural Language Generation for Spoken Dialogue System using RNN Encoder-Decoder Networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*; Association for Computational Linguistics: Vancouver, BC, Canada, 2017. [CrossRef]
43. Qun, H.; Wenjing, L.; Zhangli, C. B&Anet: Combining bidirectional LSTM and self-attention for end-to-end learning of task-oriented dialogue system. *Speech Commun.* **2020**, *125*, 15–23. [CrossRef]
44. Dai, Z.; Yang, Z.; Yang, Y.; Cohen, W.W.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv* **2019**, arXiv:1901.02860.
45. Varshney, D.; Ekbal, A.; Nagaraja, G.P.; Tiwari, M.; Gopinath, A.; Bhattacharyya, P. Natural Language Generation Using Transformer Network in an Open-Domain Setting. In *Natural Language Processing and Information Systems: 25th International Conference on Applications of Natural Language to Information Systems (NLDB 2020)*; Springer: Cham, Switzerland, 2020; Volume 12089, pp. 82–93. [CrossRef]
46. Henderson, M.; Casanueva, I.; Mrkšić, N.; Su, P.; Wen, T.-H.; Vulic, I. ConveRT: Efficient and Accurate Conversational Representations from Transformers. *arXiv* **2020**, arXiv:1911.03688.

47. Oluwatobi, O.; Mueller, E.T. DLGNet: A Transformer-based Model for Dialogue Response Generation. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020.
48. Vlasov, V.; Mosig, J.; Nichol, A. Dialogue Transformers. *arXiv* **2019**, arXiv:1910.00486.
49. Mehrjardi, M.S.; Trabelsi, A.; Zaïane, O.R. Self-Attentional Models Application in Task-Oriented Dialogue Generation Systems. *arXiv* **2019**, arXiv:1909.05246.
50. Vlasov, V.; Drissner-Schmid, A.; Nichol, A. Few-shot generalization across dialogue tasks. *arXiv* **2018**, arXiv:1811.11707.
51. Sun, P.; Yang, X.; Zhao, X.; Wang, Z. An Overview of Named Entity Recognition. In Proceedings of the 2018 International Conference on Asian Language Processing (IALP), Bandung, Indonesia, 15–17 November 2018; pp. 273–278. [CrossRef]
52. Alonso, M.A.; Gómez Rodríguez, C.; Vilares, J. On the Use of Parsing for Named Entity Recognition. *Appl. Sci.* **2021**, *11*, 1090. [CrossRef]
53. Lui, M.; Baldwin, T. Langid py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*; Association for Computational Linguistics: Jeju Island, Korea, 2012; pp. 25–30.
54. Harms, J.-G.; Kucherbaev, P.; Bozzon, A.; Houben, G.-J. Approaches for Dialog Management in Conversational Agents. *IEEE Internet Comput.* **2019**, *23*, 13–22. [CrossRef]