Department of Computer Science, University of Potsdam

Internship Title

Analysis of Intent Recognition of Modular Dialog Systems

Supervisors

Prof. Dr. Lena A. Jäger
Department of Computer Science, University of Potsdam
An der Bahn 2, 14476 Potsdam, Germany
lejaeger@uni-potsdam.de

Jan Nehring
German Research Centre for Artificial Intelligence (DFKI)
Alt-Moabit 91c, 10559 Berlin
jan.nehring@dfki.de

Submitted By

Akhyar Ahmed

Table of Contents

1. Introduction

In practical applications, Dialog systems (DS) often combines multiple DS. These systems select the appropriate sub-DS for each incoming user utterance to generate the answer for the user. Nehring and Ahmed (2021) called these systems Modular Dialog Systems (MDS). To the user, the MDS appears as a single, unified DS. There are various reasons for such a modular structure of DS: i) The designers of a DS might want to combine several existing DS and save the effort of migrating them into a joint system. For example, multiple departments create chatbots in a company, and one wants to join these chatbots together into a unified system. ii) A new DS should combine several existing DS created using different technologies and cannot be implemented in a single joint DS. Alternatively, iii) a dialog system becomes so big that the performance of its NLU decreases because the model is not suitable to handle such large amounts of data. In research, dialog systems often use machine learning for their dialog managers(DM). In industry applications dialog systems often build on a different architecture: Popular dialog frameworks like Google Dialog Flow(GDF), IBM Watson Assistant (IWA), or Rasa rely on rule-based DM. Further, they rely heavily on the concept of intent. Since no established term exists yet in the literature to the best of our knowledge, we will refer to these systems as Intent-Based Dialog Systems (IBDS). Because of the different architecture, IBDS requires another kind of dataset. IBDS usually only uses machine learning for the Natural Language Understanding component, whereas the DM is rule-based. Combining multiple IBDS into a unified architecture is a gap in the scientific literature. Although it is an issue in industry applications, see, e.g., the Google Mega Agent or the concept of Skills in IWA, its solution is not trivial. To the best of our knowledge, it has not been addressed in research yet. We formally describe the task *module selection* (MS), which brings the problem of multidomain dialog systems to IBDS. Section 4 also proposes an evaluation methodology for this task. Further, we publish a benchmark dataset for this task (Section 5.1 and 5.2) and describe its characteristics (Section 5.3). We divided the whole experiment into two-part in the first part we proposed some baseline models with loss functions and did an experiment with them. We are going to discuss these baseline models with loss functions in section 6. But it didn't work out. Then we introduce several models that provide a strong baseline for this task. In a series of experiments, we show that the text of the user utterance provides a more robust feature for models for MS than confidence values of Intent Detection (ID). Additionally, the experiments serve as a proof-of-concept of the applicability of the MS task, the evaluation methodology, and the dataset. The scientific contributions of this work are i) to formally introduce the novel task MS and an evaluation framework for MS, ii) to present a dataset for the evaluation of models for MS, and iii) to present three baseline models for MS. We iv) show in our experiments that text is a more important and reliable feature for models for MS than confidence values of ID.

2. Background

2.1. Intent-Based Dialog System

Many DS used in practical applications are based on intents. Intents are discrete labels that express what the user wants to achieve with an utterance. The designer of the dialog system defines the list of intents that the DS can understand. Usually, the DS assigns a single intent to each utterance, but DS that assigns multiple intents to an utterance also exists (Liu et al., 2019a). In this type of architecture, each user utterance is processed first by the Natural Language Understanding component (NLU). The NLU performs two main tasks: i) ID, which maps the user utterance to one intent taken from a predefined list of intent classes. Usually, an ID model predicts a label and outputs a confidence score

for the prediction. The confidence score expresses the probability that the model's prediction is correct. However, the confidence scores of modern neural networks are not always reliable (Guo et al., 2017).
ii) The second task of the NLU is Entity Recognition. The DM is usually modeled as a finite-state machine. Hand-crafted rules define how the outcome of the NLU changes the state of this finite-state machine and, therefore, the state of the dialog after each user utterance. The rules can trigger database lookups and also external API calls. The author of the dialog system predefines answers. The answers are attached to the state of the DM and therefore also determined by the rules.
Usually, IBDS is a task-oriented dialog system, meaning that they assist the user in solving a task, although they are also capable of a limited amount of chit-chat. The dialog usually revolves around finding values for slots. E.g., a chatbot that books tables for a restaurant need to fill the slot's date, number of persons, and callers' name. These systems are called *slotfillers*, and the task of determining the slot values is called *slotfilling*. Slotfiller IBDS resembles the surprisingly old GUS architecture (Bobrow et al., 1977; Jurafsky and Martin, 2021). IBDS, as defined here, is different from DS using stochastical DM, which is often based on the POMPD framework (Williams and Young, 2007), see (Young et al., 2013) for a review. Instead of relying on rule-based DM, these systems learn their DM from data. Their processing pipeline includes the additional step of dialog state tracking (DSTC) between NLU and DM, which determines the value of all slots in each dialog turn. The DM is trained using reinforcement learning. They require training data for ID and ER, similar to IBDS. Additionally, they require annotated dialogs as training data for DSTC and the DM.

2.2. The Modular Dialog System Framework

The Modular Dialog System (MDS) framework (Nehring and Ahmed, 2021) defines an architecture for combining several DS. Figure 1 shows the architecture of MDS. In this architecture, each DS is called a module. Each incoming user utterance is processed by an MS component that decides which module is appropriate to process this utterance. This module then produces the answer for the user. The MS task was first introduced as part of the MDS framework (Nehring and Ahmed, 2021). This paper extends this work, formally describes the task, and proposes a dataset and an evaluation methodology. MDS is different from other multi bot dialog system architectures because they deliberately do not use a joined system to solve a multi-task problem. Joined systems (D'Haro et al., 2015; Planells et al., 2013; Song et al., 2018) often show a better performance than modular ones. At the same time, the architecture of joined systems tends to be more complex. MDS, by contrast, can be built from existing dialog systems and require only a new MS component. The modules of an MDS do not necessarily need to be IBDS. The architecture is capable of handling other types of dialog systems also, for example, question answering (see, e.g., (Nehring et al., 2021)). However, this paper focuses on MDS consisting of multiple IBDS.

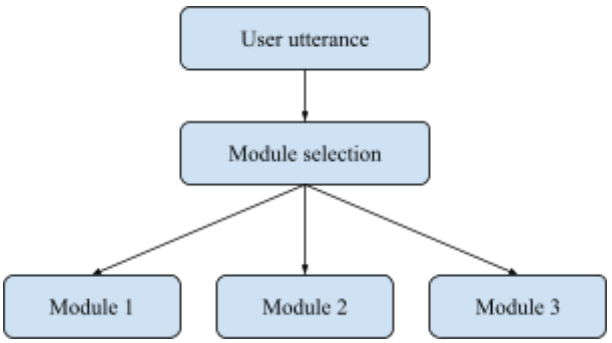

Figure 1: Architecture of modular dialog system.

2.3. Google Dialogflow, IBM Watson Assistant, and Rasa

This project uses the off-the-shelf DS GDF, IWA, and Rasa as modules to construct an MDS. All three are commercial products. GDF and IWA are closed source, whereas Rasa is open source. In GDF and IWA, authors of a dialog system define their chatbot in a browser-based user interface. In Rasa, the dialog designer works with text files and a command-line application. The processing pipelines for user utterances of all three systems are similar and follow the architecture of IBDS as defined in Section 2.1. Rasa augments the rule-based DM by machine learning to allow conversation paths that the dialog designer did not anticipate. The exact implementations of the NLU of GDF and IWA are the corporate secrets of the respective owners. Rasa uses the Dual Intent and Entity Transformer as NLU (Bunk et al., 2020).

2.4. Evaluation of Dialog Systems

There are many ways how to evaluate a DS. User studies give insights into how efficient and effective a DS assists a user in solving a particular task; see Moller (2005) for an extensive review. However, these evaluations are time-consuming and expensive. Evaluation methods typically used in computational linguistics can only examine parts of the DS, such as the ID and ER components. These are text classification problems, and precision, recall, F1 scores, and accuracy are standard metrics used to evaluate models for IR and ED. Liu et al., (2019a). Liu et al. (2019b) and Larson et al. (2019) are examples of NLU systems being evaluated using F1 scores.

3. Related Works

3.1. Combination of Dialog Systems

The scientific literature lists many approaches how to combining multiple DS. However, these approaches either do not combine IBDS, or they do not include an evaluation. A straightforward approach is to ask the user at the beginning of the dialog which domain he wants to talk about and limit the dialog to the respective sub-DS only, as in Clara (D'Haro et al., 2015). Some dialog systems do not use intents. Retrieval-based systems search an extensive database of dialog turns for a user utterance similar to a given user utterance and output the answer from the database. Generation-based dialog systems model the whole dialog system in a single neural network as a sequence-to-sequence problem through methods such as GPT2 (Radford et al., 2019). Both retrieval-based and generation-based systems produce an answer without any additional information. Song et al. (2018) and Tanaka et al. (2019) use reranking to produce an answer in a combination of such systems: Each sub dialog system produces an answer. Then, a heuristic ranks the list of answers. The system presents the highest-ranking answer to the user. The DialPort framework (Zhao et al., 2016a; Zhao et al., 2016b) connects multiple spoken dialog systems and knowledge sources. It uses a master agent that selects a DS or a knowledge base that answers a user utterance. The master agent has an NLU and a dialog state tracking component. It uses the Semi-Markov Decision Process (Sutton et al., 1999) framework for selecting the suitable agent or knowledge source that can answer the user utterance. Multidomain dialog systems are often a combination of multiple DS, one for each domain. Usually, the term multidomain dialog systems refers to dialog systems that consist of multiple sub dialog systems with stochastical DM. For an overview of the state-of-the-art, see DSCT8 (Li et al., 2020) and DSTC9 (Li et al., 2021).

3.2. Datasets

This work builds on HWU64 (Liu et al., 2019b), which is a dataset for NLU, ID, and ER. Section 5 describes this dataset in more detail. Another comparable dataset is CLINC150 (Larson et al., 2019) which contains additional out-of-scope utterances. Both datasets include several domains. By contrast, the NLU dataset Banking77 (Casanueva et al., 2020) contains many intents from a single domain only. An important dataset for multidomain dialog systems is MultiWoz (Budzianowski et al., 2018). This dataset targets DS with stochastical DM and is therefore not applicable to the MS task. Another related dataset is the Dialog Dodecathlon (Shuster et al., 2020) which measures the performance of a DS in 12 different tasks, including question answering, persona grounding, empathetic dialog, and more.

4. Module Selection: A New Task

This section formally describes the task of MS. MS is the task of assigning user utterances to modules of an MDS. Given a modular dialog system with n modules $M_{1...n}$, the MS function *MS* assigns a module M_i *(i)* to a user utterance u as shown in equation 1.

$$MS(u) \rightarrow M_i \quad (1)$$

After MS decides on a module, this module can produce the answer to the given user utterance. Different features are possible as inputs to a model that solves this task. One noticeable feature is the text of the user utterance. In this case, the task resembles domain classification. Another prominent feature is the confidence scores of the intent classification of the modules. We propose two measures to evaluate the quality of MS. First, one can directly evaluate the quality of the MS task, which is a classification task that can be evaluated using F1 scores. We call the F1-score of MS $F1_{MS}$ in the remainder of this report. The second measure is the quality of ID, which is also measured using F1 scores. We call it the F1-score of ID $F1_{ID}$. $F1_{ID}$ gives a more direct insight into the quality of the DS, while $F1_{MS}$ directly measures the quality of the MS step in the processing pipeline. For additional analysis, one can measure the precision and recall of ID and MS. Comparing the MDS to an analogous DS implemented as a single module can give an insight into whether the MDS architecture is appropriate or if it would make more sense to implement a single module DS. We call the single-module implementation the *non-modular scenario*. Accordingly, the *modular-scenario* denotes the DS distributed over several modules. It is hard to implement both systems in practical applications to gain both numbers. However, in this project, we calculated the F1 scores of ID in the non-modular scenario $F1_{ID,nonmod}$ for each of the DS Rasa, GDF, and IWA. Often intent datasets are imbalanced. Therefore, in the remainder of this paper, we use micro-F1 scores to take the class imbalance into account. Depending on the application, macro F1 scores can be a valid alternative.

5. The Dataset

As a basis for our new dataset, we chose the dataset HWU64 (Liu et al., 2019b), which is a dataset for ID And entity recognition. It contains 25,716 utterances from the home automation domain from 68 intents in 18 scenarios. One scenario is, for example, “alarm” with intents such as “set alarm”, “query alarm” and “remove alarm”. The creators did not name the dataset, but to our knowledge, it was first referred to as HWU64 by Casanueva et al. (2020).

5.1 A New Dataset for Module Selection: First Experiment

For the first experiment, we randomly subsample the dataset with their scenarios and assign them to the three connectors GFD, IWA, and Rasa. For the three dialog system connectors we created three datasets we call DS-1, DS-2, and DS-3 (Figure 2) then we assign each dataset to each connector. Each

dataset has four subsets, tr-bot: training data for the respective connectors, tr-ms: training data for our MDS, valid: validation dataset for our MDS, test: test set to test our MDS models.

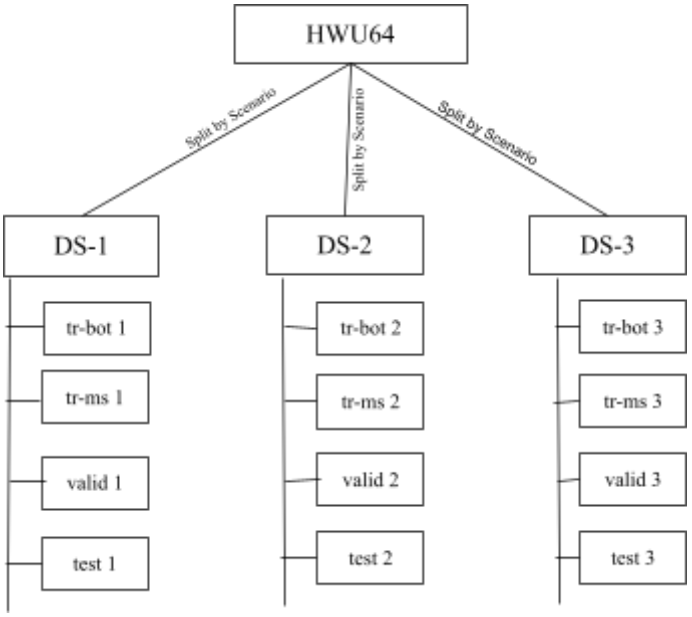

Figure 2: Sample data distribution among GDF, IBMW, and Rasa.

To get the intents and confidence values we create two agents from each dialog system. One was trained with its assigned dataset tr-bot, which we call $GDF_{self,}$ $IBMW_{self,}$ and $Rasa_{self,}$ and another agent was trained with all three tr-bot sets which we call $GDF_{all,}$ $IBMW_{all,}$ and $Rasa_{all}$ (Figure 3). Figure 3 describes this for GDF when DS-1 was assigned to GDF.

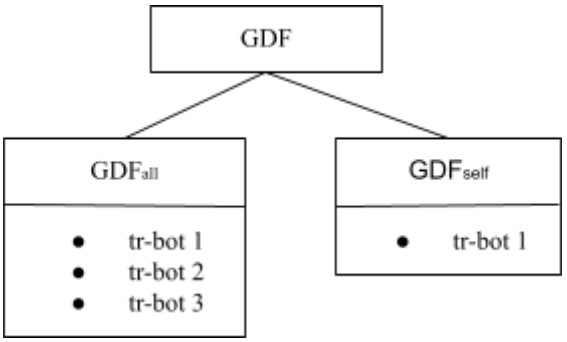

Figure 3: Sample agent distribution among each dialog system.

5.2 A New Dataset for Module Selection: Second Experiment

In the second experiment, we split the data into four equally sized parts: $train_{ID}$ is the training data for the DS' NLU. $train_{MS}$ is the training data for the MS. Finally, there is a *valid* and a *test* set. We processed all samples by the three NLUs GDF, IWA, and Rasa, and recorded the detected intent and confidence scores. We randomly assigned the scenarios to the three dialog systems GDF, IWA, and Rasa. In our experiments, we found out that both $F1_{ID,mod}$ and $F1_{MS,mod}$ can vary a lot depending on this random assignment. Therefore, we repeated our dataset creation process five times to create five splits. Further, to compute the quality of the non-modular scenario $F1_{ID,\ non\text{-}mod}$ we processed the whole dataset once with each module. Table 1 describes the columns of the MS training data.

column	description	example value
id	unique identifier	0
hwu64 id	id from the original HWU64 dataset	6962
utterance	The user utterance string	What alarms did I set today.
true intent	true label of ID task	alarm:query
scenario	scenario annotation from hwu64	alarm
split	split annotation (from 0-4)	0
target agent	target label of MS task	google dialogflow
dataset	can be train_{MS}, train_{ID}, valid or test	test
GDF intent	intent predicted by GDF	alarm:query
GDF confidence	confidence of GDF	0.78
Rasa intent	intent predicted by Rasa	datetime:query
Rasa confidence	confidende of Rasa	0.51
IWA intent	intent predicted by IWA	calendar:set
IWA confidence	confidence of IWA	0.33

Table 1: Description of the columns of the dataset.

It was expensive to process the dataset with its 25,716 utterances five times in the modular and three times in the non-modular scenario. Further, the dataset contains many samples for some intents: 1440 samples for the intent with the most samples and 25% of the intents have 623 or more samples. We find that this high number of samples is unrealistic in practical use cases. Therefore, we subsampled the dataset: In each split and each intent, we removed random samples so that each intent has a maximum of 100 samples, approximately 25 for each dataset $train_{ID,}$ $train_{MS,}$ *valid*, and *test*. We repeated this process separately over the different splits such that each of the splits contains different samples. In this way, we reduced the size of the dataset by 75.36% for each split.

5.3. Dataset Characteristics

Like the HWU64 dataset, our new dataset cannot create a fully functioning dialog system because it does not contain any data for a DM or system responses. The user utterances are not embedded in the context of the dialog. Therefore, models for MS based on this dataset cannot take the dialog history into account. Our new dataset shares this problem with NLU datasets such as HWU64, CLINC150, or Banking77. Further, we did not include the entities from HWU64 in the dataset. Table 2 shows the statistics of the dataset. It shows the number of intents, scenarios, and samples for the five splits.

split	target agent	number intents	number scenarios	number samples
0	GDF	12	4	1,200
	IWA	25	7	2,216
	Rasa	31	7	2,921
1	GDF	21	5	1,930
	IWA	30	10	2,791
	Rasa	17	3	1,616
2	GDF	13	4	1,300
	IWA	26	6	2,330
	Rasa	29	8	2,707
3	GDF	20	5	1,921
	IWA	31	7	2,905
	Rasa	17	6	1,511
4	GDF	15	4	1,413
	IWA	26	6	2,513
	Rasa	27	8	2,411

Table 2: Number of intents, scenarios, and samples for the different splits of the dataset.

Figure 4 shows boxplots of the confidence values of the three DS over the five splits. The table shows confidence values for both in and out of domain samples for each DS. One can see that the average in-domain confidence values differ between the DS. In general, Rasa produces higher confidence values than the others. Across all dialog systems, out-of-domain samples can reach confidence as high as 1.0.

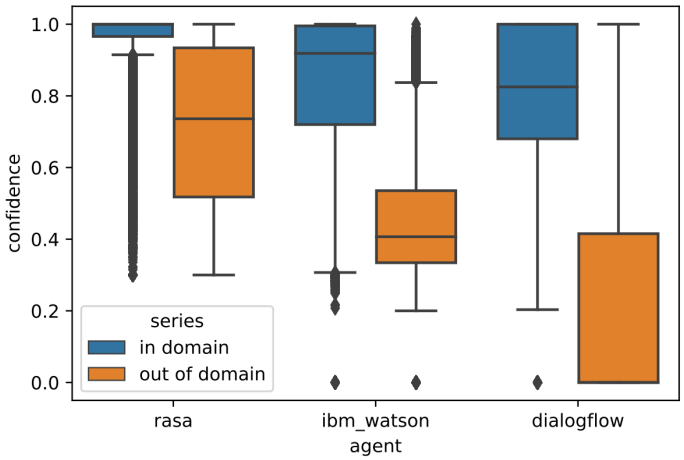

Figure 4: Distribution of confidence values for Google Dialogflow, Rasa, and IBM Watson Assistant for in domain and out of domain samples.

Table 3 shows the performance of the dialog systems in the non-modular scenario. A side result of our experiments is the comparison of ID of GDF, IWA, and Rasa (Table 3. IWA has the best-performing ID on our dataset. Liu et al. (2019b) reported similar results, although, in their experiment, the differences between GDF, IWA, and Rasa were less significant.

Module	$F1_{ID,nonmod}$	$P_{ID,nonmod}$	$R_{ID,nonmod}$
GDF	0.78	0.78	0.78
IWA	0.86	0.86	0.86
Rasa	0.75	0.75	0.75

Table 3: F1-score, precision (P), and recall (R) of the three dialog systems in the non-modular scenario.

6. Model Architectures and Evaluation

This section proposes three model architectures to solve the module selection task. We evaluate the models using our proposed dataset and evaluation methodology.

6.1 Model Architectures

We use the modular dialog system from our dataset with $n = 3$ modules for all models. For each utterance u, which is of the string value, we generate a vector $x \in R^n$ of confidence values. In a real-world system, one would send each incoming user utterance to the modules and collect the confidence values of the modules' ID. In our experiments, we can use the confidence values from our dataset. Each model outputs a vector $y \in R^n$. We encode the labels in y as one-hot-encoding, meaning that each position in y corresponds to a label (GDA, IWA, or Rasa). The position of the output vector with the highest activation determines the model's prediction. All three models use linear layers: A

linear layer is a linear transformation $y = xA^T + b$. $x \in R^n$ is the input vector, $y \in R^n$ is the output vector, $b \in R^n$ is the bias vector, and $A \in R^{n \times n}$ is a matrix. The dimensionality of x, A, and b can vary.

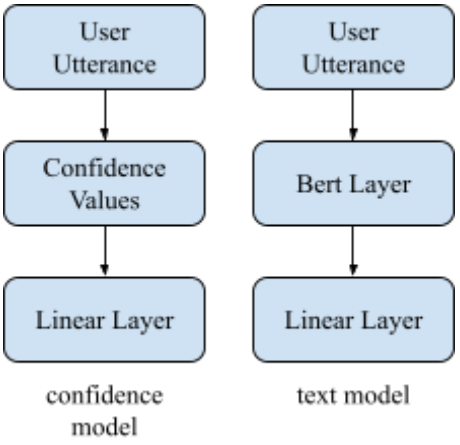

Figure 5: Architecture of the *confidence* and *text* models.

The *confidence* model (Figure 5) uses confidence values x_C as features for the model. A linear layer maps the confidence values x_C to the output layer. The trainable parameters of the model are A and b of the linear layer. Model *text* (Figure 5) uses the user u utterance as a feature. It uses a standard BERT for sequence classification architecture (Devlin et al., 2019) which uses a linear layer on top of a pre-trained BERT model. The trainable parameters are the parameters of the BERT model and parameters A and b of the linear layer.

6.1.1 Baseline Models: First Experiment

In the first part of this analysis, mainly we proposed two baseline models on the basis of their loss functions and normalization methods. Each model is a joint model of the *confidence* model and *text* model. And for the *confidence* model, there is a normalization layer added on the top of the linear layer. The first one is a model with a single loss we named it *baseline* model (Figure 6).

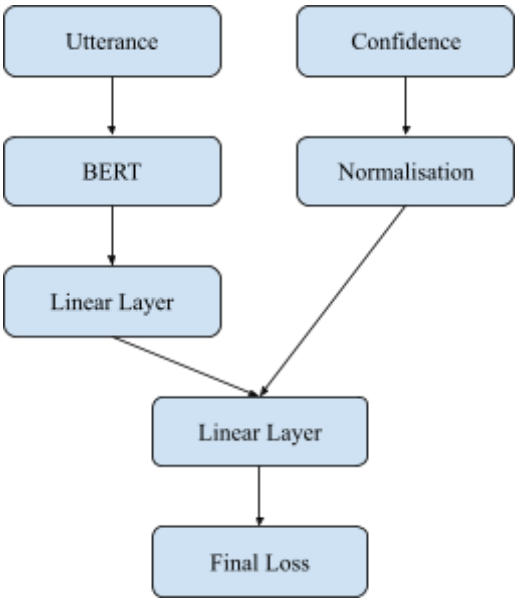

Figure 6: Architecture of the *baseline* model with a single loss.

From the left side and the right side of this model it respectively takes user utterance *u* and confidence value x_c. User utterance *u* then send to a pre-trained BERT model and confidence x_c went through a normalization method. The model gives *n* length of output from both sides. Then it concatenates both *n* length of outputs and calculates the final loss $loss_{baseline}$ (equation 2) using another linear layer.

$$loss_{baseline} = 1/n \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \qquad (2)$$

Here *n* is the number of modules, y_i is the true value for module *i* for one sample (0 or 1 or 2), and $\hat{y}_i$ is the predicted value for module *i* for each sample.

From this *baseline* model, we created four more baseline models with a combination of different normalization functions. The first one is *baseline model 1* it uses no normalization method on its confidence model. The second model is *baseline model 2*, it uses softmax (equation 3) normalization methods to normalize the confidence values for each sample.

$$softmax(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \qquad (3)$$

The third model is *baseline model 3*, here temperature scaling Muzafari et. al.(2019) (equation 4) was used as the normalization method to normalize confidence values.

$$\sigma_{temperature\_scaling}(z_i, T) = softmax(\frac{z_i}{T}) \qquad (4)$$

Finally the fourth model of baseline models with single loss is called *baseline model 4* and it uses a simple normalization term from Nehring and Ahmed, 2021 paper. The trainable parameters for all four baseline models are the BERT layer and the two linear layers.

The second baseline model is called the baseline model with triple loss (Figure 7). The base of this model is almost identical to the baseline model with a single loss model instead it calculates loss three times.

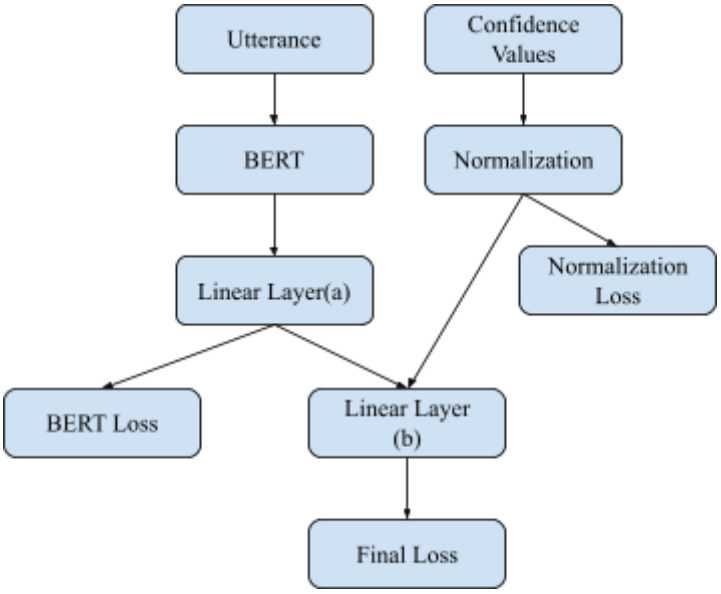

Figure 7: Architecture of a *Baseline model with triple loss.*

The first loss was calculated after getting the output from the linear layer of the *text* part of the model, Second loss comes from the output of the *confidence* part of the model, and then finally calculate a final loss from the last the output of concatenated linear layer and merge the previous two losses which are $loss_{tripleloss}$ (equation 5).

$$loss_{triplemodel} = 1/n \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + 1/n \sum_{i=1}^{n} (y_i - \hat{a}_i)^2 + 1/n \sum_{i=1}^{n} (y_i - \hat{c}_i)^2 \quad (5)$$

Where, n is the number of modules, y_i is the true value for module i for one sample (0 or 1 or 2), $\hat{y}_i$ is the predicted value for module i for one sample, $\hat{a}_i$ is the output of linear layer *(a)* for module i, $\hat{c}_i$ is the output of normalization layer for module i.

6.1.2 Baseline Models: Second Experiment

Model *text + confidence* (Figure 8) is a combination of the former two models. In the figure, the left branch is the confidence model and the right is the text model. Both produce an output of length n, which the model concatenates to a vector $x^2 \in R^{2n}$. x^2 is then mapped to y using another linear transformation. The trainable parameters are the BERT layer and the three linear layers. This trained on the $train_{MS}$ section of the dataset using Cross-Entropy Loss (Bishop, 2006).

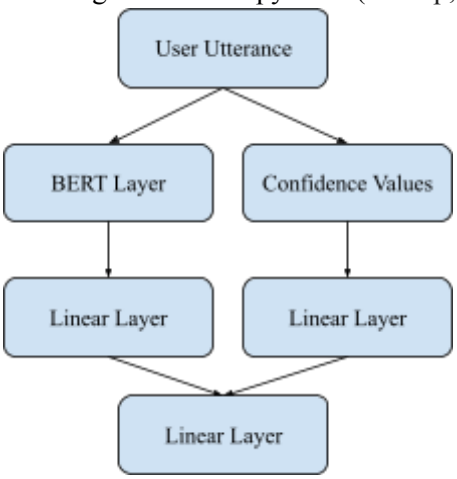

Figure 8: Architecture of the confidence + text model.

6.1.3 Baseline models: Hyperparameters

Training examples are encoded in one-hot encoding, meaning that they are *n-dimensional* vectors with 0 on every position, except for the position of the label, which is 1. We perform a grid search using the *valid* dataset to find optimal values for batch size (32) and learning rate ($5e^{-5}$). Metrics $F1_{ID}$ and $F1_{MS}$, along with respective precision and recall values, are calculated on the *test* part of the dataset. We train and evaluated the models for each split and averaged the metrics.

6.2 Results

6.2.1 Results: First Experiment

Tables 4 and Figure 9 show the results of the evaluation of the modular scenario for MS (Table 4).

Model	$P_{MS,mod}$	$R_{MS,mod}$	$F1_{MS,mod}$
Triple loss model	0.95	0.95	0.95
Baseline Model-1 Features:text,confidence, and no normalization	0.94	0.94	0.94
Baseline Model-2	0.90	0.92	0.92

Features:text, confidence, and Softmax normalization			
Baseline Model-3 Feature:text, confidence, and Temperature Scaling	0.92	0.93	0.93
Baseline Model-4 Features:text, confidence, and simple normalization from Nehring and Ahmed, 2021	0.94	0.93	0.94
Confidence Model	0.72	0.72	0.72
Text Model	0.94	0.94	0.94

Table 4: F1-score (F1), precision (P) and recall (R) of module selection of all seven models in the modular scenario.

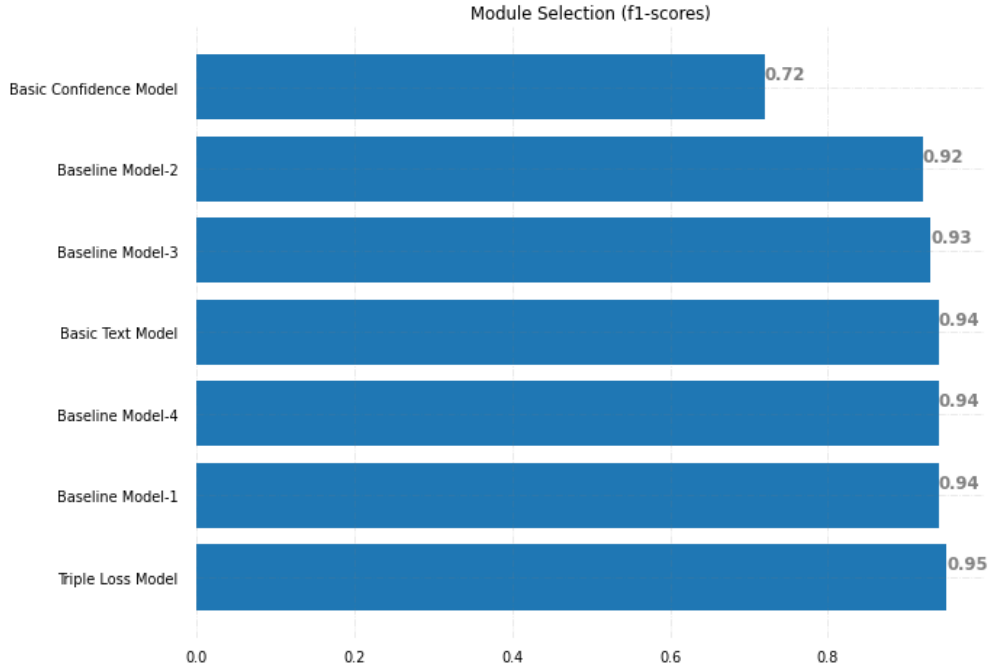

Figure 9: Shows the comparison of f1-scores of module selection for all seven models.

Tables 5 and Figure 10 shows the results of the evaluation of the modular scenario for MS (Table 5).

Model	**$P_{ID,mod}$**	**$R_{ID,mod}$**	**$F1_{ID,mod}$**
Triple loss model	0.82	0.82	0.81
Baseline Model-1 Features:text,confidence, and no normalization	0.80	0.81	0.80

Baseline Model-2 Features:text, confidence, and Softmax normalization	0.82	0.82	0.80
Baseline Model-3 Feature:text, confidence, and Temperature Scaling	0.81	0.81	0.80
Baseline Model-4 Features:text, confidence, and simple normalization from Nehring and Ahmed, 2021	0.81	0.82	0.80
Confidence Model	0.74	0.72	0.71
Text Model	0.82	0.82	0.81

Table 5: F1-score (F1), precision (P) and recall (R) of ID using all seven models in the modular scenario.

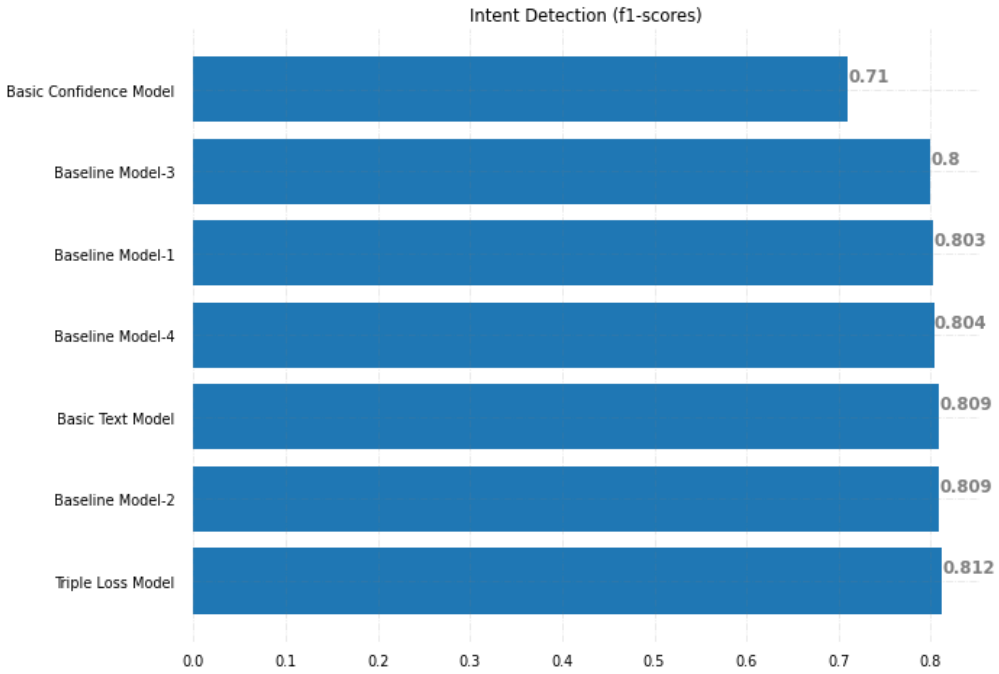

Figure 10: Shows the comparison of f1-scores of intent detection for all seven models.

6.2.2 Results: Second Experiment

Tables 6 and 7 show the results of the evaluation of the modular scenario for MS (Table 6) and ID (Table 7). The metrics for models text and *text* + *confidence* are much higher than the scores of the confidence model. The scores of *text* + *confidence* are marginally higher than *text* only.

Model	**$P_{MS,mod}$**	**$R_{MS,mod}$**	**$F1_{MS,mod}$**
conf	0.34 (0.08)	0.31 (0.09)	0.28 (0.09)
text	0.89 (0.02)	0.89 (0.02)	0.89 (0.02)
text + *conf*	0.91 (0.00)	0.91 (0.00)	0.91 (0.00)

Table 6: F1-score (F1), precision (P), and recall (R) of module selection of the three models in the modular scenario. We averaged the scores over the five splits. The number in brackets is the standard deviation. We abbreviated the word *confidence* with *conf* in the model name for brevity.

Model	$P_{ID,mod}$	$R_{ID,mod}$	$F1_{ID,mod}$
conf	0.35 (0.08)	0.25 (0.08)	0.24 (0.08)
text	0.78 (0.02)	0.74 (0.01)	0.75 (0.01)
text + conf	0.79 (0.02)	0.75 (0.01)	0.76 (0.01)

Table 7: F1-score (F1), precision (P), and recall (R) of ID using the three models in the modular scenario. We averaged the scores over the five splits. The number in brackets is the standard deviation. We abbreviated the word *confidence* with *conf* in the model name for brevity.

6.3 Discussion of the experiments

From the first experiment Table 4, 5, and Figure 9, 10 shows that the best confidence was achieved by our novel neural architecture which combined textual and confidence features. We also showed that text is a much stronger feature than confidence values in our setting. In our experiments, normalization functions did not influence the performance. Though *baseline model with the triple loss* model shows very promising improvements to all other baseline models. In the baseline model with the triple loss model we just simply merge all the losses from the textual model and confidence model. We don't know the proper weights for these losses. Moreover, we rerun the whole experiment again by taking different samples (e.g., 5, 10, and 50) of the training set. That time the *triple loss* model didn't show the best performance. So that's why we didn't step forward with the *baseline model with the triple loss* model. Proper Gridsearch is needed to find the weights of all losses for the *baseline model with the triple loss* model.

In the second experiment Table, 6 shows that *text* is a much more informative feature for module selection than the confidence of the underlying models. The confidence model has a low $F1_{MS}$ of 0.28. This results in a low $F1_{ID}$, making it unusable in practical applications. Comparing $F1_{ID,\ mod}$ (Table 7) and $F1_{ID,nonmod}$ (Table 3) shows that the performance of the modular scenario is comparable to the performance of the non-modular scenario for the model's *text* and *text* + *confidence*.

7. Conclusion

We have presented *module selection* as a novel task. Further, we have presented a dataset to evaluate module selection models and an evaluation framework. We have presented three models and evaluated them using the dataset and the evaluation methodology. We have shown that text is, at least in our dataset, a more reliable feature for module selection than confidence scores. The models serve as a strong baseline for future work in module selection. Our presented evaluation framework does not directly evaluate the quality of the DS. Instead, it measures the quality of ID in the modular scenario. Like NLU datasets like HWU64, CLINC150, or Banking77, it does not take dialog history into account. Dialogs often span more than one turn, and neglecting the dialog context is a critical limitation. Nevertheless, we argue that this evaluation framework is a practical way of evaluating the quality of module selection in modular dialog systems.

8. Acknowledgments

This work was partially supported by the German Federal Ministry of Education and Research under grant 01|S20043 (Lena A. Jager).

9. References

[1] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). "On Calibration of Modern Neural Networks." *34th International Conference on Machine Learning,* ICML 2017, 3:2130–2143, jun.

[2] Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H. S., and Winograd, T. (1977). GUS, A Frame-Driven Dialog System. *Artificial Intelligence*, 8(2):155–173.

[3] Nehring, J. and Ahmed, A. (2021). Normalisierungsmethoden fur Intent Erkennung Modularer Dialogsysteme. In Benjamin Weiss Stefan Hillmann, editor, *Tagungsband der 32. Konferenz. Elektronische Sprachsignalverarbeitung (ESSV-2021), March 3-5, Berlin, Germany*. TUDpress

[4] Nehring, J., Feldhus, N., Kaur, H., and Ahmed, A. (2021). Combining Open Domain Question Answering with a Task-Oriented Dialog System. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 38–45, Online, Aug. Association for Computational Linguistics.

[5] Williams, J. D. and Young, S. (2007). Partially observable Markov decision processes for spoken dialog systems. *Computer Speech Language*, 21(2):393– 422.

[6] Young, S., Gasi˘ c, M., Thomson, B., and Williams, ´ J. D. (2013). POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proceedings of the IEEE*, 101(5):1160–1179.

[7] D'Haro, L. F., Kim, S., Yeo, K. H., Jiang, R., Niculescu, A. I., Banchs, R. E., and Li, H. (2015). CLARA: A Multifunctional Virtual Agent for Conference Support and Touristic Information. In *Natural Language Dialog Systems and Intelligent Assistants*, pages 233–239. Springer International Publishing, Oct.

[8] Planells, J., Hurtado, L.-F., Segarra, E., and Sanchis, E. (2013). A Multi-domain Dialog System to integrate heterogeneous Spoken Dialog Systems. Technical report.

[9] Song, Y., Li, C.-T., Nie, J.-Y., Zhang, M., Zhao, D., and Yan, R. (2018). An Ensemble of Retrieval-Based and Generation-Based Human-Computer Conversation Systems. *In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4382–4388. International Joint Conferences on Artificial Intelligence Organization.

[10] Bunk, T., Varshneya, D., Vlasov, V., and Nichol, A. (2020). DIET: Lightweight Language Understanding for Dialogue Systems. *arXiv*.

[11] Moller, S. (2005). *Quality of Telephone-Based Spoken Dialogue Systems*. Kluwer Academic Publishers, Boston.

[12] Liu, J., Li, Y., and Lin, M. (2019a). Review of Intent Detection Methods in the Human-Machine Dialogue System. *Journal of Physics: Conference Series*, 1267:12059, Jul.

[13] Liu, X., Eshghi, A., Swietojanski, P., and Rieser, V. (2019b). Benchmarking Natural Language Understanding Services for building Conversational Agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, Ortigia, Siracusa (SR), Italy, apr. Springer.

[14] Larson, S., Mahendran, A., Peper, J. J., Clarke, C., Lee, A., Hill, P., Kummerfeld, J. K., Leach, K., Laurenzano, M. A., Tang, L., and Mars, J. (2019). An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*

(EMNLPIJCNLP), pages 1311–1316, Hong Kong, China, November. Association for Computational Linguistics.
[15] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners.
[16] Tanaka, R., Ozeki, A., Kato, S., and Lee, A. (2019). An Ensemble Dialogue System for Facts-Based Sentence Generation. arXiv.
[17] Zhao, T., Lee, K., and Eskenazi, M. (2016a). DialPort: A General Framework for Aggregating Dialog Systems. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pages 32–34, Austin, TX, November. Association for Computational Linguistics.
[18] Zhao, T., Lee, K., and Eskenazi, M. (2016b). DialPort: Connecting the spoken dialog research community to real user data. In 2016 *IEEE Spoken Language Technology Workshop (SLT)*, pages 83–90.
[19] Sutton, R., Precup, D., and Singh, S. (1999). Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. *Artificial Intelligence*, 112:181–211.
[20] Li, J., Peng, B., Lee, S., Gao, J., Takanobu, R., Zhu, Q., Huang, M., Schulz, H., Atkinson, A., and Adada, M. (2020). Results of the Multi-Domain TaskCompletion Dialog Challenge. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, Eighth Dialog System Technology Challenge Workshop*.
[21] Li, J., Zhu, Q., Luo, L., Liden, L., Huang, K., Shayandeh, S., Liang, R., Peng, B., Zhang, Z., Shukla, S., Takanobu, R., Huang, M., and Gao, J. (2021). Multi-domain Task-oriented Dialog Challenge II at DSTC9. In *AAAI-2021 Dialog System Technology Challenge 9 Workshop*.
[22] Casanueva, I., Temcinas, T., Gerz, D., Henderson, M., and Vulic, I. (2020). Efficient Intent Detection with Dual Sentence Encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online, July. Association for Computational Linguistics.
[23] Budzianowski, P., Wen, T.-H., Tseng, B.-H.,Casanueva, I., Ultes, S., Ramadan, O., and Gasic, M. (2018). MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium, October-November. Association for Computational Linguistics.
[24] Shuster, K., Ju, D., Roller, S., Dinan, E., Boureau, Y., and Weston, J. (2020). The Dialogue Dodecathlon: Open-Domain Knowledge and Image Grounded Conversational Agents. In Dan Jurafsky, et al., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2453–2470. Association for Computational Linguistics.
[25] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). 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),* pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
[26] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.
[27] Mozafari, A., Gomes, H., Leão, W., Janny, S., Gagné, C. (2019). Attended Temperature Scaling: A Practical Approach for Calibrating Deep Neural Networks. arXiv.